

[Journal](#)
[home page](#)
[General](#)
[information](#)
[Submit an](#)
[item](#)
[Download](#)
[style files](#)
[Copyright](#)
[Contact us](#)

THE PracTeX Journal



Issue Table of Contents

2006, Number 3

[Published 2006-08-15]

Notices

[From the Editor:](#) In this issue

Dave Walden

[Feedback](#)

From the Readers

[Report on PracTeX06](#)

From the Participants and Compiled by Dave Walden

[Whole Issue PDF for PracTeX Journal 2006-3](#)

The Editors

Articles

[Productivity with macros and packages](#)

Will Robertson

[Replacing LaTeX2e standard classes with KOMA-Script](#)

Yuri Robbers, Markus Kohm and Rasmus Pank Roulund

[The memoir class](#)

Peter Wilson

[LaTeX curricula vitae with the CurVe class](#)

Didier Verna

[ConTeXt Starters](#)

Joe Hogg

[Creating Pocket-size Books Using LaTeX](#)

D.V.L.K.D.P. Venugopal

Columns

[Travels in TeX Land: Experiences Refining Page Layout for a Book](#)

David Walden

[Book Review: *Formatting Information -- A beginner's introduction to typesetting with LaTeX* by Peter Flynn](#)

D.V.L.K.D.P. Venugopal

[Ask Nelly:](#)

What are the differences among MiKTeX, ProTeXt, and LaTeX?

How do I add change bars?

The Editors

[Distractions: LaTeX Wordplay — 3 crosswords; math font quiz answers](#)

The Editors

[Journal](#)
[home page](#)
[General](#)
[information](#)
[Submit an](#)
[item](#)
[Download](#)
[style files](#)
[Copyright](#)
[Contact us](#)

THE PracTeX Journal



From the Editor: In this issue

Dave Walden

- [Comment on this paper](#)
- [Send submission idea to editor](#)

Hello,

My name is Dave Walden, and I am the guest editor of this issue of *The PracTeX Journal* (TPJ). I am on the Editorial Board of the journal, but until now I have worked in the journal's back room maintaining the journal's web pages.

In this issue

I was hoping that this issue would have a theme of "using TeX to improve productivity and using TeX more productively," and two or three pieces in the issue are explicitly on that topic (for instance, [pracjournal/2006-3/robertson/](#)). However, as I look at the whole collection of papers and columns in this issue, I realize that a key to both TeX's productivity and using it productively is its *flexibility*. In fact, flexibility may be the underlying theme of every issue of TPJ and may help distinguish TeX from standard word processing systems and other typesetting systems -- TeX is extraordinarily flexible and customizable in many dimensions.

Users of TeX have several formats they can choose from: plain TeX, LaTeX, ConTeXt ([pracjournal/2006-3/hogg/](#)), and so forth.

Users have different major classes they can choose from: basic LaTeX, KOMA ([pracjournal/2006-3/robbers/](#)), memoir ([pracjournal/2006-3/wilson/](#)), AMS, and so forth.

Typesetting all manner of languages is possible. There was great excitement at the recent PracTeX06 conference ([pracjournal/2006-3/practex06/](#)) about XeTeX's capability (along with the fontspec package from our editorial board member Will Robertson) to *easily* use any font in a Mac, Windows, or Linux font folder (including non-Latin fonts).

TeX can be used any type of application as illustrated by various articles in this issue: creating a pocket-size book

([pracjournal/2006-3/venugopal-pocketbook/](#)), developing a letterhead or a laying out a small brochure ([pracjournal/2006-3/hogg/](#)), laying out a book ([pracjournal/2006-3/walden/](#)), or formatting crossword puzzles ([pracjournal/2006-3/distract/](#)).

All this capability comes with lots of free documentation, e.g., [pracjournal/2006-3/venugopal-review/](#).

As usual, this issue of TPJ includes some other standard features: letters from readers ([pracjournal/2006-3/readers/](#)), and Nelly's answers to questions ([pracjournal/2006-3/asknelly/](#)), and a PDF of the whole issue ([pracjournal/2006-3/wholeissue/](#)).

Thanks

Helping with reviewing and copy editing for this issue were Barbara Beeton, Karl Berry, Jon Breitenbucher, Lance Carnes, Peter Flom, David Ignat, Reinaldo Opus, John O'Rourke, Yuri Robbers, and Will Robertson (please forgive me if I have forgotten anyone). Especially notable is the work of Yuri Robbers who solicited two of the papers for this issue, edited one of them, led the writing of a third paper, and helped with production editing. Will Robertson (who also wrote a paper) and Lance Carnes (who was supposed to be taking a break) also did extensive production editing. Other members of our editorial board also commented on the content of this issue in various useful ways. Thank you to all of them and to the authors.

Next issue

Editor Lances Carnes states that next issue will include papers from the recent PracTeX06 conference, but will also be accepting other papers; so if you have something to say about the world of TeX, please write it down and send it to this journal.

Page generated August 19, 2006 ; [TUG home page](#); [search](#); [contact webmaster](#).

[Journal](#)
[home page](#)
[General](#)
[information](#)
[Submit an](#)
[item](#)
[Download](#)
[style files](#)
[Copyright](#)
[Contact us](#)

THE PracTeX Journal



Feedback

From the Readers

- [Comment on this paper](#)
- [Send submission idea to editor](#)

TPJ

How does one join the TeX user group?

[Please see tug.org/join.html for information and a link to the membership form. -Ed.]

TPJ

Dear editor,

I am a beginner in using LaTeX to write a paper. Will you please give me an example of preparing a journal paper (double spaced, one column, 12pt, etc).

Thanks a lot in advance!

Regards, Haixia Zhang

[Editor's note: Several members of our editorial board discussed this request. There were a few main thoughts:

- Many journals already have their own LaTeX style. Google on "LaTeX journal style" to find several examples. The first thing Haixia Zhang should do is ask if the journal he has in mind has an existing style file.
- Double-spaced is a style that was used with "traditional" typewritten submissions. The LaTeX journal styles that we know of don't use this.
- An interesting question worthy of a future TPJ journal article is how to maximize portability among

journal styles as an author repeatedly submits his or her paper until a journal accepts it.

If you would like to write an article on the subject of the last point above, please contact the editors at pracjournal@tug.org.]

TpJ

Dear Sir/Madam,

I very much liked the article by Stephen Hartke on Free Math Fonts [in issue 2006-1 (tug.org/pracjournal/2006-1/hartke/)]. It seems like almost all the fonts have only LaTeX support and not Plain TeX support. Would it be possible for those of us who prefer Plain TeX to prevail upon the font authors to make header files such that we Plain TeX-ies can use their Math Fonts too?

This is even more necessary now that LaTeX has moved forward and offers limited support for LaTeX 2.09 and earlier. And an article by Michael Spivak in your journal showed that there are people who don't wish to use LaTeX, for whatever reasons.

Sincerely yours, Jesse Deutsch

[A response from Karl Berry: (1) Plain TeX has no standard macros for font support, so it is not entirely clear what to provide. (2) Font authors are almost certainly not going to be able to help address the writer's desire, as they are busy making fonts and many of them probably do not use plain TeX. However, they would probably welcome such a contribution. If there are certain font families of particular interest, please email pracjournal@tug.org and we can see if a plain TeX volunteer can be found to work on it.]

TpJ

[Editor's note: the following note is in reference to the third Ask Nelly question in issue 2006-2 (tug.org/pracjournal/2006-2/asknelly/).]

Dear Sir,

I believe that the mdwlist package has options for suspending and resuming list environments. Maybe is worthwhile to mention it in your column.

Yours sincerely, Rafael Pappalardo

TpJ

[Editor's note: the following notes is in reference to the article by D.V.L.K.D.P. Venugopal in issue 2006-2 (tug.org/pracjournal/2006-2/asknelly/).]

Dear Sir,

Very good. An excellent way to structure a lecture series.

Sincerely, Keith Jones

TpJ

[Editor's note: we are unsure which article in issue 2006-2 the following note refers to. We tried to clarify it but without success. We guess it refers to the article by Hlavacek on Ipe (tug.org/pracjourn/2006-2/hlavacek/).]

Dear Sirs,

This article really helped me out with the different image possibilities with LaTeX under MS Windows.

But I still had problems finding a suitable MS Windows tool to incorporate .eps until I found and installed The Gimp for MS. It would be helpful if you could update section 5 of your paper by indicating that there is a version of The Gimp for Windows!

Regards, Laurent Peckels

TpJ

Issue 2006-2 of *The PracTeX Journal* reprinted a an opinion piece from *OSNews* by Andy Roberts (tug.org/pracjourn/2006-2/roberts/), and the reprint included responses to the author after the piece's original publication.

[*OSNews reader*:] For horizontal spacing issues, however, you have the reverse problem: LaTeX would rather overrun a right margin than leave too much space between words. (The infamous "overfull hbox", whose black slug indicating an error certain styles remove, incidentally, even in draft mode... grrrr) I'm not sure why Knuth thought an overrun was such a better idea than extra whitespace in an hbox, while extra white space was preferable to an overrun in a vbox, but the result in many published papers, and even some books, has been ugly. It certainly does not look professional, but the only way to fix it is to do some really obscure TeXing, or else completely rephrase your wording (the universal fallback I've seen in all LaTeX manuals).

I don't see how one would term

```
\begin{sloppypar}
Stuff which one wants typeset in a fashion which will allow ugliness....
\end{sloppypar}
```

as "really obscure", or just use `\sloppy` (hopefully with a matching `\fussy`).

[*Another OSNews reader*:] It is insanely difficult to get text paragraphs to flow around

arbitrarily shaped inserts.

Try using `\usepackage{shapepar}` and draw up the desired shape in xfig.

[*OSNews reader:*] For ultra-high-end artistic typesetting (think of Bringhurst's *Elements of Typographic Style*), LaTeX is probably not the best tool, although with some of the experimental micro-typography additions to pdfTeX it is 99% of the way there.

Actually, pdfTeX would've typeset that *much* better, and Peter Wilson's memoir class shows as an example how to achieve that pagestyle -- I still find it reprehensible that it's advocated hanging punctuation since the first edition (though that index entry vanished from the second) yet only used it on the back cover (of the second edition).

I'm also endlessly amused by Quark and InDesign users who decry my usage of TeX or LaTeX, claiming that features I find essential or useful are not important -- until Quark or InDesign gains said feature at which point they crow about it endlessly.

The person wanting OLE TeX equations could look into how Mac OS X and one of the Equation Service programs interact -- should be possible to achieve something like to that in Windows.

William Adams
Mechanicsburg, PA

TpJ

[From Tom Colson regarding the Andy Roberts LaTeX article (tug.org/pracjourn/2006-2/roberts/)]

I would have graduated a year earlier had I read this....

TpJ

[Journal home page](#)
[General information](#)
[Submit an item](#)
[Download style files](#)
[Copyright](#)
[Contact us](#)

THE PracTeX Journal



Report on PracTeX06

From the Participants and Compiled by Dave Walden

- [Comment on this paper](#)
- [Send submission idea to editor](#)



This journal is a child of the PracTeX conference series; it was conceived two years ago at the second PracTeX conference in San Francisco. As this journal was pulling its seventh quarterly issue together, the third PracTeX06 conference was held at Rutgers, the State University (Busch Campus), Piscataway, New Jersey, USA.

The conference began with a four day LaTeX workshop followed by a three day conference with a mostly idle day in between.

[Editor's note: For this conference report, I solicited two or three sentences from each participant on his or her impressions of the conference generally, of one specific session, and one sentence on any other aspect of PracTeX06. Several attendees followed the "rules" and submitted the requested two or three sentences, a couple of attendees said they could (and did) provide sentences on more presentations as needed, and the enthusiastic and prolific Boris Veytsman sent remarks on *all* presentations except the *three* he was involved in giving. Although my intention was to select only one comment for each presentation, in the end I included everything I received.]

General comments on the conference

Kaveh Barzagan: Content was possibly the best ever. Precisely relevant to Practical TeX, and a minimum of geekspeak. All presenters did a great job. The location was disappointing with nothing much to do around the hotel.

Barbara Beeton: The enthusiasm reminded me of early TUG annual meetings, which tells me that TeX's time isn't over yet. And the informal

networking that everybody indulged in is perhaps the most important aspect of a meeting like Practical TeX, as well as one of the most enjoyable.

Karl Berry: As an organizer of the conference, I was very happy that the general consensus seemed to be that nearly all the presentations "provided useful information, rather than being over my head." (Several people used essentially that phrase.)

Elizabeth Dearborn: I truly enjoyed participating in PracTeX 06, with so many knowledgeable, interesting people, and I look forward to PracTeX07.

Peter Flom: The conference was excellent, with many informative talks at different levels. I would encourage more attendance (and participation) by people who are relatively new to the world of TeX -- the presenters are friendly and the water is fine.

Troy Henderson: I got more out of this conference than I have from any other conferences that I've attended.

Robin Laakso: I thought the talks and interaction at this PracTeX conference were particularly rewarding, inspiring and fun. Sometimes it seemed like a real-life version of a support mailing list ... which is the whole point of organizing a Practical TeX conference, so yay :)

Donald Pellegrino: "Practical TeX 2006" was a great opportunity to see how TeX is being used by the experts who really know how to use it.

Boris Veytsman: This was a very pleasant and useful meeting. The atmosphere there was just right.

Dave Walden: This was my first TeX or TUG conference; I learned much, was inspired to try several new things, and loved meeting all the TeX users



Many (not all) conference participants posing for a photo

Detailed conference agenda

Tuesday, July 25, through Friday, July 28: the LaTeX workshop with volunteer instructors Sue DeMeritt and Cheryl Ponchin.

Saturday, July 29: Many people arrived by plane, train, and car for the conference proper. An informal reception was held from 5-7pm on Saturday evening at the Clarion Hotel in Edison, NJ, where many conference attendees stayed. At the reception, conference attendees made new acquaintances and renewed old acquaintances.

The conference itself was held on Sunday, July 30, through Tuesday August 1, at the Core Building on the Rutgers (Busch) campus. Each day consisted of two morning sessions separated by a break, lunch, and two afternoon sessions separated by a break. Other formal and informal events were fit in before, after, and between formal sessions.

Sunday, July 30

9am Karl Berry -- Opening & introductions

9:20am Barbara Beeton -- keynote address: How to Create a TeX Journal: A Personal Journey

Steve Grathwohl: It was a wonderful presentation, full of stuff I didn't know and was glad to learn.

Boris Veytsman: The keynote address was an interesting and enjoyable journey into the history of TeX, TUG, *TUGboat*.

Dave Walden: Barbara gave a fascinating tour through the various changes in the style of *TUGboat* over the years. Even more exciting was hearing and meeting, Barbara in person; she has been the editor of *TUGboat* for essentially its whole life, and *TUGboat* in turn was the initial reason I joined TUG.

10:20am break

10:30am Peter Flom -- LaTeX for social scientists and other people who think they don't need it

Robin Laakso: Peter's funny, common sense perspective on encouraging people to use (La)TeX was instructive and personally inspiring. He ended his talk with a "wish list" of suggestions on how to improve its ease of use. It was amazing, and at times amusing, to observe Peter's wish list getting smaller and smaller as attendees recommended solutions and gave advice.

Boris Veytsman: Peter Flom's presentation gave me a rare opportunity to see how TeX is viewed by a relative novice; after all these years of working with TeX I sometimes lose this fresh view.

11:10am Jim Hefferon -- LaTeX resources

Karl Berry: Jim's tour of LaTeX resources operated was at a level we need more of -- not just saying you can use x, y, or z, but actually making recommendations and choices.

Boris Veytsman: Jim Hefferon's work gave a bunch of references to LaTeX resources; there were some new points for everybody, and I picked up a couple too.

11:50am Boris Veytsman -- Design of presentations: Notes on principles and TeX implementation

Dave Walden: Taking off from Edward Tufte's 31-page, self-published essay, *The Cognitive Style of PowerPoint*, in which Tufte denigrates PowerPoint style presentations, particularly their bulleted lists, Boris gave many illustrations of another approach to presentations; he particularly emphasized the use of the pdfscreen program (from attendee Kaveh Bazargan's company) to avoid the artificial boundaries that each PowerPoint slide tends to put into the flow of a presentation. (Boris's admonition to avoid bullets on slides became a running joke for all the rest of the presenters at the conference.)

12:30pm lunch

1:40pm Alan Wetmore -- TeX and after dinner speaking

Boris Veytsman: Alan Wetmore's presentation gave me many new ideas about interactive texts and presentation and how to make them myself.

2:20pm Steve Peter -- Fonts, typefaces, glyphs & sorts

Boris Veytsman: Steve Peter's beautiful talk was a fascinating representation of the history of typography.

3pm break

3:10pm Klaus Höppner -- Creation of a PostScript Type 1 logo font with MetaType1

Karl Berry: I found Klaus's tutorial on using MetaType1 in practice very carefully worked out and quite entertaining to boot.

Boris Veytsman: Klaus Hoepfner's talk explained techniques for MetaPost/MetaType1 work; it was a useful exposition of a rather obscure material.

3:50pm William Adams -- TypeSpec v2: typesetting font specimens

Boris Veytsman: William Adams's talk was a very passionate showcase of great fonts; I really enjoyed the beauty of typography.

4:30pm Q&A session

5pm Four parallel birds-of-feather sessions

- Professional Publishing and Meeting Journal Guidelines for PDF with TeX
- Maintaining a large TeX installation
- Customizing TeX Shop
- Trial run slide show introducing (La)TeX
- Additionally, a MacTeX group discussion took place at the hotel Saturday afternoon; they also met for lunch on Sunday.

Monday, July 31

9am Ned Hummel -- Common macro pitfalls and how to avoid them

Boris Veytsman: Ned Hummel's talk was a very good introduction into logical markup and an exposition of the sad fact that most people choose the (seemingly) least resistance path of visible markup - and are bitten hard down the road.

9:40am Jonathan Kew -- XeTeX: past, present, future

Kaveh Bazargan: For me, XeTeX was the star of the show, bringing simplicity to TeX.

Troy Henderson: Of course, the talk on XeTeX was particularly interesting because of the ease of adding and using new fonts.

Donald Pellegrino: The presentation not only helped me understand what XeTeX is, but it also helped me understand how silly I am for not using it already.

Dave Walden: In addition to describing XeTeX, Jonathan repeatedly mentioned the value of Will Robertson's fontspec package for use in conjunction with XeTeX.

Boris Veytsman: Jonathan Kew brilliantly presented his great work on XeTeX -- a plot to make the TeX font guru obsolete.

[Editor's note: I have allowed a disproportionate number of comments on this presentation. As Kaveh says above, it was the "star of the show"; this status was clear from the spontaneous burst of applause from the participants when Jonathan dragged a new (non-TeX-format) font file to his computer's fonts directory and, with no other machinations, his document compiled correctly using the new font.]

10:20am break

10:30am Federico Garcia -- Capabilities of PDF interactivity

Boris Veytsman: The talk by Federico Garcia gave a very good short introduction into PDF capabilities of hyperref -- something interesting for various users.

11:10am Boris Veytsman, Maria Shmlevich -- Automatic report generation with web, TeX and SQL

Dave Walden: This presentation showed how a small number of weeks of work was able to produce a system that automatically formatted reports required under U.S. government contracts based on minimal essential input from the people doing reporting.

11:50am Kaveh Bazargan -- Removing vertical stretch - mimicking traditional typesetting with TeX

Boris Veytsman: Kaveh Bazargan presented both an elegant solution to a very old problem (typesetting to a grid), and a very interesting talk about his company, which represents a way to go in introducing TeX to real life environments.

12:30pm lunch

1:40pm David Walden -- A lifetime as an amateur compositor



The amateur (your editor for this issue) lectures his betters

Karl Berry: I found Dave's review of pre-computer technologies, with props, a whole lot of fun!

Peter Flom: I particularly liked the talk by Dave Walden on some tips for writing a book using LaTeX -- I have already implemented some of them.

Boris Veytsman: Dave's fascinated me by its historical part: it put the digital typography in a perspective.

2:20pm Troy Henderson -- Using MetaPost for creating high-quality technical graphics

Robin Laakso: I enjoyed Troy's talk on MetaPost very much. I thought the department logo he re-created with MetaPost looked great, and was wowed by his LaTeX and MetaPost previewer, both of which could be handy to use at various venues.

Boris Veytsman: The two presentations about graphics (by Troy Henderson and by Andrew Mertz and William Slough) were a useful introduction into great and useful graphics packages.

3pm break

3:10pm Andrew Mertz, William Slough -- Graphics with PGF and TikZ

Karl Berry: I thought Andrew and William did an excellent job presenting a graduated series of examples with TikZ, a friendly front-end to PGF (I gather), for producing graphics entirely within TeX. (They did a similarly excellent job introducing the beamer package last year.)

3:50 Jon Breitenbucher -- LaTeX at a liberal arts college

Barbara Beeton: Jon Breitenbucher's presentation on tactics for getting students not only to use TeX, but to improve the quality of their content while doing so was full of ideas that could profitably be used to spread the word.

Boris Veytsman: We all could easily relate to the talk by Jon Breitenbucher about successes and tribulations in teaching LaTeX to students.

4:30pm Q&A; TUG meeting

Tuesday, August 1

9am Peter Flynn -- Rolling your own document class

Elizabeth Dearborn: It is extremely difficult to choose only one presentation to comment upon, but Peter Flynn's talk on rolling your own document class has inspired me to try something I thought was beyond my capabilities.

Boris Veytsman: The talk by Peter Flynn was a good introduction for fledgling class writers.

9:40am Federico Garcia -- LaTeX and the different bibliography styles

Boris Veytsman: The talk by Federico Garcia described, among other things, his great work on the very useful opcit package. It was interesting & practical.

10:20am break

10:30am Boris Veytsman, Leila Akhmadeyeva -- Drawing medical pedigree trees with TeX and pstricks

Dave Walden: We learned a lot about certain kinds of genealogical tree diagrams used in medical research and diagnosis and how TeX can support creating them; the insight I gained surely applies to other application areas.

11:10am Elizabeth Dearborn -- TeX and medicine

Boris Veytsman: Elizabeth Dearborn's talk was one of the most inspiring presentations at the conference. The details of book publishing and the dos and don'ts of the process were quite interesting.

11:50am Bob Neveln, Bob Alps -- Writing and checking complete proofs in TeX

Boris Veytsman: The presentation by Bob Neveln and Bob Alps gave an insight into the math papers of the future, when computers will check not only our spelling, but the soundness of our logic as well (God forbid they do the same with the speeches of our politicians!)

12:30pm lunch

1:40pm Stephen Moye -- A wayward wayfarer's way to TeX

Boris Veytsman: Stephen Moye's talk was a great talk about the life with TeX by a wise man and a very good storyteller.

2:20pm Steve Peter -- Introduction to memoir

Boris Veytsman: This presentation by Steve Peter was a good introduction into the capabilities of memoir.

3pm break

3:10pm panel -- Barbara Beeton, Peter Flynn, Mirko Janc, Jonathan Kew; moderator: Dave Walden

Boris Veytsman: I really liked the panel; the questions were interesting, and the possibility for all of us to talk about the future of TeX was priceless.

4pm closing

7pm banquet at the Clarion hotel

Kaveh Bazargan: Perhaps one alcoholic drink could have been provided per person, with the rest available at the bar.

Wednesday, August 2: Visit to New York City and the Morgan Library.

Karl Berry: The Morgan Library was awe-inspiring. To see an actual Gutenberg Bible, especially, along with so many other exemplars of the finest printing, made me remember why I was drawn into this world in the first place.



Conference organizers Karl Berry and Robin Laakso with Dante president Klaus Höppner at the top of the Empire State Building

[Journal](#)
[home page](#)
[General](#)
[information](#)
[Submit an](#)
[item](#)
[Download](#)
[style files](#)
[Copyright](#)
[Contact us](#)

THE PracTeX Journal



Productivity with macros and packages

Will Robertson

Abstract

LaTeX's advantages in productivity, for me, are due to its ability to be customised. The first half of this article discusses small macros written to ease document production, with some examples of how I use macros to save time and effort. Then, I briefly cover a selection of packages that provide a whole heap of functionality that other people have kindly implemented.

Will Robertson is a PhD student in Mechanical/Mechatronic Engineering in the University of Adelaide, South Australia. His LaTeX experience came about from being distracted with it while writing various reports and papers on his "real" research, and now he's hooked. Will may be contacted at `will@guerilla.net.au`

- [PDF version of paper](#)
- [Comment on this paper](#)
- [Send submission idea to editor](#)

Page generated August 19, 2006 ; [TUG home page](#); [search](#); [contact webmaster](#).

Productivity with macros and packages

Will Robertson

Email `wspr81@gmail.com`

Address School of Mechanical Engineering, University of Adelaide, Australia

Abstract L^AT_EX's advantages in productivity, for me, are due to its ability to be customised. The first half of this article discusses small macros written to ease document production, with some examples of how I use macros to save time and effort. Then, I briefly cover a selection of packages that provide a whole heap of functionality that other people have kindly implemented.

1 Introduction

One of L^AT_EX's advantages lies in its separation of content and formatting. This concept, called logical markup, enables authors to write without the distraction of worrying about typesetting. Of course, authors aren't *forced* to work this way, but it's highly recommended.

Implicit in the idea of logical markup is the ability to define your own logical elements as the text requires. These are known in L^AT_EX as macros or commands, and can be as simple as `\newcommand\strong[1]{\textbf{#1}}` to define a command for **strongly** emphasising words, as so:

```
...for \strong{strongly} emphasising...
```

A whole world opens up when you can write your own macros; in my own documents, whenever there's an issue of formatting that will re-occur, it gets a macro in case I change my mind later. But when things start to get more complex, there are better ways to do things—it's much better to use other people's solutions; *i.e.*, other people's packages.

This article is split into two halves. The first half, section 2, discusses small macros written to ease document production, with some examples of how I (personally) use macros to save time and effort. In the second half, section 3 on page 9, I briefly cover a bunch of packages that provide a whole heap of functionality that other people have kindly implemented—no need to do it yourself!

Formatting in this article Verbatim in this document is represented by typewriter text in [bright blue](#). Packages & classes are typeset in a sans serif font, ‘hyper-linked’ to their CTAN documentation (where available; a couple won’t work at time of publication but should in the future), and coloured [dark red](#) like the other external links in the document. Internal links within the document are coloured [green](#).

2 Writing your own macros

Writing things like `Figure~\ref{fig:myfigure}` every single time a figure is referred to results in errors creeping in that are hard to detect; typical of such errors might include misspelling ‘Figure’, the absence of the non-breaking space (the tilde ~, something even experienced L^AT_EX users often forget to use), and so on. The command `\newcommand` becomes a new friend to relieve these problems. Suddenly, it’s easy to define commands such as `\figref` that will expand out to the figure string and the reference:

```
\newcommand\figref[1]{Figure~\ref{fig:#1}}
```

After adding this command to the preamble, I can now write `\figref{myfigure}` to refer to my figures. Two advantages are gained with this method: error-checking (the output will now always be consistent and correct or throw an error) and flexibility. If we wish to change our string to look like ‘Fig. 7’, it’s a matter of changing one line in the preamble:

```
\newcommand\figref[1]{Fig.\,\ref{fig:#1}}
```

(Note the thin space `\,`, which is appropriate in this case after the period and which is also often omitted — to the detriment of the output quality.)

While you can use the above macro in your documents, there is a macro package [refstyle](#) (see page [12](#)) that does the job a lot more completely. It includes macros for all sorts of references (tables, sections, footnotes, *etc.*) and even allows you to reference more than one figure at a time, such as

```
\figref{myfig1,myfig2,myfig3}
```

Figures 1, 2 and 3

2.1 How *not* to use macros

I have now briefly introduced macros as a way to be productive. In the section after this one (page 5), I'll continue along these lines. But it is said that a little bit of knowledge can be dangerous. It's worth considering how macros should not be used. The following is a good template for inserting figures:

```
\begin{figure}[htbp]
  \centering
  \includegraphics{*}
  \caption{*}
  \label{fig:*}
\end{figure}
```

In fact, in my editor (TeXShop for Mac OS X), I've set it up so that dragging an image into a source document inserts this snippet automatically with the filename and label filled in with the path and name of the graphic respectively. All I have to write is the caption itself.

This isn't an ideal situation, however. Using such a template 'freezes' the formatting decisions at the instant it's inserted; if changes are desired on a large scale, repetitive search-and-replace is required. It is better to start to use macros that define the formatting from a single place. Here is an example of one of my first efforts. Coincidentally, it's quite similar to an example given by Dave Walden [3]:

```
\newcommand\insertfig[2]{%
  \begin{figure}[htbp]
    \centering
    \includegraphics{#1}
    \caption{#2}
    \label{fig:#1}
  \end{figure}}
```

At first, the attraction is clear. Isn't the following much easier to write?

```
\insertfig{filename}{This is an example figure caption.}
```

This version has much to offer, in that the formatting may now be adjusted from a single place in the document — a feature always to be aspired to. But we quickly run into trouble.

Firstly, semantics have been lost. The arguments to the command are unnamed and arbitrary. Consider a smart editor which parses the document for `\label` commands and builds a list from which to prompt for `\ref` instances. How will it know that `fig:figname` is a label to use?

Secondly, we begin to want to add more functionality. Say we want a short caption (for headers/footers and the Table of Contents) and a long caption (for the actual text). Well, easy. Just change the macro to accept an optional argument, just like the `\caption` command does. Okay, now let's say we want to stick in some graphics options, such as scaling. More work, more arguments. And don't forget the option to change the figure placement from `[htbp]`, which should always be considered in the fine-tuning of the typesetting of the completed document.

Now our previous calls with this command, throughout the entire document, need to be amended to take the other argument into account. This might not be that much work with a fancy search 'n' replace, but this is supposed to be 'L^AT_EX for productivity', right?

That was a long winded way of saying that the approach taken above is too coarse-grained for our application. As a rule of thumb, restrict new commands to performing a single action, to avoid such problems. In the above example, the only formatting is to centre the figure on the page. The following redefinition of the `figure` environment provides this:

```
\renewenvironment{figure}[1][\fps@figure]
  {\@float{figure}[#1]\centering}
  {\end@float}
```

Because we're overwriting the existing definition of the `figure` environment, it's necessary to use `\renewenvironment` instead of `\newenvironment`. `\fps@figure` is the default float placement (usually `tbp`), and the `\@float... \end@float` pseudo-environment is the internal L^AT_EX generalised method for creating floats.¹

1. I should warn some of you that if this code is pasted into a regular document (not your own package), it must be preceded by `\makeatletter` and ended with `\makeatother`. More details can be found in Robin Fairbairns' T_EX FAQ answer '`\@` and `@` in macro names'.

2.2 Examples of my macros

Two sections ago, I discussed how macros should be defined for error-checking, flexibility, and consistency. In the previous section, I showed that macros should be as specific as possible; in general they shouldn't be used as a method for simplifying the input as an end to itself. Below, I'll give some examples of macros (and the types of macros) I often use.

2.2.1 Foot or margin (or end) notes

The first is a macro for adding notes to text. Generally, this refers to footnotes² but sometimes one might wish to annotate their texts in a slightly more interesting manner (\rightarrow). Or even using end notes, as discussed by Dave Walden [4]. Semantically, these could be exactly the same thing, so the markup for denoting this should be the same. Therefore,

As a more interesting example.

```
\newcommand\note[1]{\unskip\footnote{#1}}
```

Note the `\unskip`. This is included so that the note needn't be placed with no whitespace preceding it in the source. For example, `some text \note{A note.}` will appear as 'some text³'. Note the omission of the space before the superscript. Now, to adjust this definition to send the note out to the margin instead, it's simply a matter of a new definition:

```
\newcommand\note[1]{%
  \unskip~\marginpar{\hspace{0pt}\raggedright\small\itshape #1}%
  ($\rightarrow$)}
```

...or however. The `\unskip~` is a nice trick to ensure that there's never a line break before the note call-out in the text;⁴ and the `\hspace{0pt}` is to ensure hyphenation in case of a long first word in the margin note.

In this case, the `\note` command adds flexibility to the document (it's easy to later change the definition to adjust the typesetting of the notes), and to enforce consistency in the output by controlling how the space around the note callout (be it superscript or other) behaves.

2. Such as this one. 3. A note.

4. I also use this in a redefinition of `natbib`'s `\cite` command when using numerical references.

2.2.2 Abbreviations

The use of Latin abbreviations (*e.g.*, *i.e.*, *cf.*, *etc.*) in formal text isn't always encouraged. Nonetheless, they can be handy, and it's important to remember how they should be punctuated. Macros can address both of these issues by easily being able to switch out the abbreviations if necessary or ensure that the punctuation is always correct. Let's begin with a simple macro to mark up words in another language; *e.g.*, `\foreign{a priori}`.

```
\newcommand\foreign[1]{\emph{#1}}
```

Since 'e.g.' and 'i.e.' will generally always be followed by a comma, it's possible to define a macro to ensure this comma isn't omitted:⁵

```
\newcommand\ensurecomma{%  
  \@ifnextchar,{ }\@latex@error{Don't forget the comma!}{}}}
```

Now, it's simply a matter of defining our abbreviations:

```
\newcommand\eg{\foreign{e.g.}\ensurecomma}  
\newcommand\ie{\foreign{i.e.}\ensurecomma}  
\newcommand\cf{\foreign{cf.}\@}
```

The `\@` ensures that no extra space is added after the period as it would if the period ended a sentence. These macros in use:

<code>I concur; \ie, I agree</code>	<code>I concur; i.e., I agree</code>
<code>Add lots of sour; \eg, five lemons</code>	<code>Add lots of sour; e.g., five lemons</code>
<code>Use white sugar; \cf\ brown sugar</code>	<code>Use white sugar; cf. brown sugar</code>

I usually also define the uppercase variants `\Eg`, *etc.*

Now here's some more definitions along these lines. Abbreviations such as 'etc.' and 'et al.' may occur at the end of sentences, so it would be unfortunate to insert the period incorrectly in these cases. Here we go:

```
\newcommand\ensuresingleperiod{\@ifnextchar.{ }\@}\@}  
\newcommand\etc{\foreign{etc}\ensuresingleperiod}  
\newcommand\etal{\foreign{et al}\ensuresingleperiod}
```

5. Inside a package, `\PackageError` would be better to use than `\@latex@error`.

These are used similarly:

Sentence ending, \etc.	Sentence ending, <i>etc.</i>
riverrun, \etc, livvy.	riverrun, <i>etc.</i> , livvy.
As discussed by Robertson \etal.	As discussed by Robertson <i>et al.</i>

Of course, many people will argue against emphasising these abbreviations with italics, while others will disagree with using the abbreviations at all. The poet E. E. Cummings might have recommended omitting all of the punctuation. The definitions above can be adjusted appropriately to suit such requirements and make the appropriate changes throughout the entire document.

In this application, using macros for the abbreviations ensures both consistency and flexibility with the formatting, not to mention that using the macros precludes spelling errors in the output without an error in the compilation — provided they are used exclusively!

2.2.3 Punctuation

Many people, now, have written macros for inserting smart dashes into their documents. I copied the TUGboat macro [1] in this regard when writing the template for this very journal.⁶ But I’m going to add a twist at the end, so keep reading.

There is more than meets the eye when using dashes in text—like this one. According to the tradition followed, differing amounts of space are used around dashes. From no space—like this—to a full-width space — such as here. In all cases, it’s desirable to avoid line breaks before them, although a break *after* — like here — is fine. Others will like to use an en-dash instead – depending on the taste of the typographer and the font being used.

These points aren’t what you want to think about while writing, and it’s easy to get things wrong. An ideal case for a macro, then:

```
\DeclareRobustCommand\dash{%  
  \unskip\nobreak\thinspace\textemdash\thinspace\ignorespaces}
```

This is the definition straight from the `pracjourn` class, and incorporates all of the details described above. The amount of surrounding space is customisable, and

6. The `pracjourn` class can be found at: <http://tug.org/pracjourn/styles/latex/>

the `\unskip/\ignorespaces` arrangement ensures consistency no matter how the macro is used in the text.

Here's a couple of small details to make things even better. First, we don't want to use the `\dash` definition above when writing things like PDF bookmarks, where plain text is the order of the day. So if the `hyperref` package is being used, the following line substitutes an ASCII dash in such cases:

```
\pdfstringdefDisableCommands{\renewcommand{\dash}{ - }}
```

Secondly, it can be a little distracting having to literally write '`\dash`' all the time in the source. Those of us lucky enough to be using unicode-aware editors might prefer to use a literal em-dash in our source to denote a text dash, naturally enough. After `\usepackage[utf8]{inputenc}` in the preamble, it is then possible to bind the meaning of `\dash` above to a UTF-8 em-dash in the source with the following incantation:

```
\DeclareUnicodeCharacter{2014}{\dash}
```

A snippet of the source document could then *look* like 'this is a dash — in the source', but it would be typeset according to all the rules given above. I consider this a great advantage for readable source!

2.3 Where to keep your macros

I generally use macros on a per-document basis. Others may prefer to keep their macros all together in one place. Both have their advantages and disadvantages. Since each document requires different logical markup, I write and adapt my macros as I go, copy and pasting from previous documents.

Alternatively, it can be very convenient to write a private package incorporating all of your own macros. This can be as simple as starting a new file `mymacros.sty` with the line

```
\ProvidesPackage{mymacros}[2006/08/15 v0.1 My custom macros]
```

Placing this file in the 'local' `texmf` tree of your distribution⁷ then provides a com-

7. E.g., `~/Library/texmf/tex/latex` on Mac OS X, `C:\localtexmf\tex\latex` on Windows, or `/usr/local/texmf/tex/latex` on Linux.

mon location to maintain your macros, accessible with `\usepackage{mymacros}` in your documents.

I don't use this method personally because I'm scared of backwards compatibility problems, although with enough foresight this shouldn't be a problem in general. Furthermore, this method assumes you want the same output from your macros in each and every document, and I don't always find this to be the case.

Further information on L^AT_EX class and package writing can be found in the documentation file `clsguide`, which can be found in `$TEXMF/doc/latex/base/` or by using the command line: `texdoc clsguide`.

2.4 Mini summary on macros

Bear in mind I've only touched the surface of how macros can be used to help you write documents more efficiently. I haven't even mentioned the `xspace` package, which simplifies how L^AT_EX commands deal with following space; `\TeX the Book` and `\TeX\ the Book` would produce identical output—no need to worry about how spaces are gobbled:

```
\let\oldTeX\TeX
\renewcommand\TeX{\oldTeX\xspace}

\TeX the Book; the \TeX{}book      TEX the Book; the TEXbook
```

When writing your own macros, be as fine-grained as possible; a macro should only do a single thing, ideally, and if more is required then write two macros. This was exemplified when I discussed wrapping up a figure environment into a single macro—in my opinion, more trouble than it's worth!

The macros shown in this section are supposed to simply be indicative of how *I* use macros. The sky's the limit when writing your own documents. Be creative! Just remember, when writing your own documents, if you find yourself writing repetitive formatting commands, a macro would make it more convenient.

3 Using others' macros in packages

At some stage in the development of learning L^AT_EX, we've learned how to write our own shortcuts and definitions well enough that we're no longer intimidated by delving into other people's packages and classes to see what's going on. I don't

know about you, but I was pretty pretty happy with myself that to customise captions, say, I understood that all I had to do was edit the following from the article class:⁸

```
\long\def\@makecaption#1#2{%
  \vskip\abovecaptionskip
  \sbox\@tempboxa{#1: #2}%
  \ifdim \wd\@tempboxa >\hsize
    #1: #2\par
  \else
    \global \@minipagefalse
    \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
  \fi
  \vskip\belowcaptionskip}
```

So if I needed, say, an en-dash instead of a colon after ‘Figure’ or ‘Table’ in my captions, it was simply a matter of copying the above and changing the appropriate parts to `#1~--~#2`.

I used to do things this way, until I came to the realisation that I was wasting my time. Why? Other people had already solved to problems I was having. What happens if we want to start using the `hyperref` package? Our macro redefinition spoils everything, because now we can’t link to the figure caption. Similar problems arise adding hyperlink support to the `\figref` command from the introduction. The point is that many packages are designed to fit in around each other, and doing this is not always a straightforward task when you’re trying to hack your own support.

Things work fine doing simple things, and for one-off solutions sometimes it can be quicker to hack your way to output that looks right. This is an oft-repeated criticism of L^AT_EX in general: it’s so much easier in Plain T_EX to do this-or-that with a quick macro (re)definition. What is wrong with these sorts of ideas it that L^AT_EX isn’t complicated because its authors wanted to obfuscate their work; rather, various functionality has worked its way into its facets that cover edge cases you’re not even considering when trying to hack your own way.

8. To make such changes, you’d look through the class and copy/paste the snippets you were interested in modifying to the preamble of your document (see footnote (1) on page 4) or your own class file. The article class can be found at `$TEXMF/tex/latex/base/article.cls`, with documentation at `CTAN:macros/latex/base/classes.dtx`.

3.1 Choosing classes and packages

In the previous sections, I've shown two tiny sets of macros that make my writing more productive. Perhaps they've inspired you along similar lines. But in many cases, it's not the best thing to do, to build up your own macro packages from scratch, because other people will have done it before. Don't waste time replicating the work of smarter and/or more experienced people!

There are a huge number of packages on CTAN which are conveniently organised by category in the T_EX catalogue: <http://texcatalogue.sarovar.org/bytopic.html>. There's a very high chance that something you wish to do is contained somewhere within. But where to start looking?

First off: choose your class. For beginners, I highly recommend an 'all-in-one' class such as `memoir` (well summarised in this issue of The PracT_EX Journal [5]) or one of the KOMA classes (also featuring in this issue [2]). These have the huge advantage of a single reference. If the formatting requires adjustment, simply search through the manual. There's not much of an easier way to get started on your own with L^AT_EX.

These integrated classes are more resistant, however, to being amended with packages. And in some cases, single-purpose packages provide more features. These days, I use `memoir` for large documents and article with packages for smaller things.

To follow is an incomplete, subjective list of packages that shouldn't be overlooked. I've left out the ones that everyone knows, such as `geometry` and `hyperref`. In the spirit of the article, you shouldn't take my word for it, but look into any other options available and see if there are alternatives that are better (for example, I've frequently heard the `typearea` package is easier in many cases than `geometry`, but I must admit to have failed to investigate it).

Note that there are a few packages for formatting section titles and contents tables (`titlesec`, `sectsty`, `titletoc`, `tocloft`) and I'm not qualified in all of them enough to give a firm recommendation on their use. Personally, I've enjoyed the simplicity of the `sectsty` packages, but it is very limited.

<code>booktabs</code>	For great tables. The manual is worth reading for its advice alone.
<code>caption</code>	Customise the formatting of captions used for figures, tables, and anything else.
<code>csquotes</code>	Error-checking and flexible quotation markup. E.g., "quoted text"

	can be typeset with quotation symbols chosen in the preamble, such as ‘ ’ or “ ” or « ». This is useful not just for multilingual documents—it also helps with ensuring quotation consistency.
bigfoot	Does absolutely everything related to footnotes.
enumitem	Provides almost everything one could want for formatting lists (such as the one you’re reading).
fixltx2e	Fixes and adds many little details. Better to be safe than sorry!
mathpazo	Don’t miss out on the <code>[sc,osf]</code> options to activate better kerning & real small caps, and old-style (lowercase) numbers. Also see the FPL Neu font, which provides further improvements with maths support soon to come.
microtype	Enables pdfTeX’s microtypographical features: margin kerning (active for this document) and font expansion (gives better output but slower processing and greater file sizes—best for print work).
natbib	Formatting citations in either author/year or numerical format. The hypernat package is essential when using numerical citations to allow the <code>sort&compress</code> feature to work in conjunction with hyperref .
pdfcolmk	Fixes colour in pdfTeX, which can sometimes break (half a page of coloured text for no reason, for example).
refstyle	The most convenient and flexible method for cross referencing, as discussed in section 2. Also uses the varioref package for smart ‘on the following page’-type functionality.
SIstyle	The most convenient way to typeset numbers with units. <i>E.g.</i> , instead of <code>1.2\times 10^{-3}\,N\{\cdot\}m</code> , just write <code>\SI{1.2e-3}{N.m}</code> .
subfig	For creating sub-tables and figures.
textpos	For putting things anywhere on the page, even using absolute positioning (<i>e.g.</i> , 3 cm from the top of the page, 4 cm left of the margin).
xcolor	Very flexible colour functionality.
zref	When it’s released, looks to be amazingly comprehensive: reference anything.

3.2 Finding packages

There's little advice I can offer on how to find out about packages that have the functionality that you're looking for. Every time I look, there's more on CTAN⁹ and the T_EX Catalogue¹⁰ that I've never seen before. Various esoteric things, such as sorting index entries from within T_EX itself (see Kees van der Laan's **BLUe format**), have been implemented ten years ago! (I hope to look into this format in more detail in the future.)

Reading `comp.text.tex`, I am regularly surprised by the breadth of packages that already exist. When looking specifically, I can only recommend exploration of CTAN, and asking questions of people who have done similar explorations themselves.

4 Summary

Formatting shouldn't be hard-coded into a document; macros to ensure flexibility and consistency are preferable. There are many packages available that provide such functionality, along with a plethora of other convenient customisations. These should be experimented with!

References

- [1] Robin Fairbairns. The new (L^AT_EX 2_ε) TUGboat macros. *TUGboat*, 17(3):282–288, September 1996. ISSN 0896-3207. URL <http://www.tug.org/TUGboat/Articles/tb17-3/tb52guid.pdf>.
- [2] Yuri Robbers, Markus Kohm, and Rasmus Pank Roulund. Replacing L^AT_EX 2_ε standard classes with koma-script. *The PracT_EX Journal*, 3, 2006. URL <http://tug.org/pracjourn/2006-3/robbers>.
- [3] David Walden. Travels in T_EX land: L^AT_EX for productivity in book writing. *The PracT_EX Journal*, 2, 2006. URL <http://tug.org/pracjourn/2006-2/walden>.
- [4] David Walden. Travels in T_EX land: Final layout of a book. *The PracT_EX Journal*, 3, 2006. URL <http://tug.org/pracjourn/2006-3/walden>.
- [5] Peter Wilson. The `memoir` class. *The PracT_EX Journal*, 3, 2006. URL <http://tug.org/pracjourn/2006-3/wilson>.

9. <http://tug.ctan.org/> 10. <http://texcatalogue.sarovar.org/>

[Journal](#)
[home page](#)
[General](#)
[information](#)
[Submit an](#)
[item](#)
[Download](#)
[style files](#)
[Copyright](#)
[Contact us](#)

THE PracTeX Journal



Replacing LaTeX2e standard classes with KOMA-Script

Yuri Robbers, Markus Kohm and Rasmus Pank Roulund

Abstract

KOMA is a complete replacement of the standard LaTeX2e classes. It is aimed more at European typography, but is easily configurable. Some of KOMA's extensions and ways to configure document layout are also available in other document classes, such as the LaTeX2e standard. This paper discusses just a few parts of KOMA, especially those that have to do with page layout and with writing letters.

Yuri Robbers holds a degree in Animal Behaviour and is a teacher, a researcher and a published author. He's always had a keen interest in typography, possibly because his father is a professional typographer. Ever since he discovered LaTeX in 1995, he's always done his best to avoid using word processors, and he has embarked on a quest to learn as much as possible about TeX and its derivatives and to apply this knowledge whenever possible. Contact him at yuri.robbers@gmail.com

Markus Kohm worked, after finishing his degree in Computer Science, as a software developer for several years, eventually as a project leader. Nowadays he takes pleasure in running the household and being a father. His interest in typography already started in secondary school, through literature. In his spare time he has been working as a typographer, typesetter, LaTeX-consultant and class author for several years already. He has worked with LaTeX since the early nineties. KOMA-Script is just the tip of the iceberg that has been created out of his many years of helping out in TeX newsgroups and mailing lists and his profession. Markus Kohm can be reached at komascript@gmx.info

Rasmus Pank Roulund was born in 1986. In the summer of 2006 he graduated from a Danish gymnasium (see: http://en.wikipedia.org/wiki/Gymnasium_school), and currently works as delivery man and warehouseman for a local medicine distributor. Next year he will study math, economics or social studies at Aalborg University. Although vanilla LaTeX is nicer than any Word document he almost instantly switched to KOMA-Script. He can be contacted at rasmus_pank@yahoo.dk

- [PDF version of paper](#)
 - [Comment on this paper](#)
 - [Send submission idea to editor](#)
-

Page generated August 19, 2006 ; [TUG home page](#); [search](#); [contact webmaster](#).

Replacing $\text{\LaTeX}2_{\epsilon}$ standard classes with KOMA-Script

Yuri Robbers, Markus Kohm and Rasmus Pank Roulund

Email yuri.robbers@gmail.com
Website <http://www.komascript.de/>

Abstract KOMA is a complete replacement of the standard $\text{\LaTeX}2_{\epsilon}$ classes. It is aimed more at European typography but is easily configurable. Some of KOMA's extensions and ways to configure document layout are also available in other document classes, such as the $\text{\LaTeX}2_{\epsilon}$ standard classes. This paper discusses just a few parts of KOMA, especially those that have to do with page layout and with writing letters.

1 Introduction

KOMA-Script is a complex bundle of classes intended to replace the $\text{\LaTeX}2_{\epsilon}$ standard classes. The standard look and feel of KOMA-Script is rather different from the standard classes and is inspired by European typographical standards. Since the whole design philosophy of KOMA-Script is new, it offers all the commands that are offered by the standard classes (except for the `letter` and `slides` class) but not vice versa. Generally speaking the KOMA classes offer far more options for adaptation and configuration than standard $\text{\LaTeX}2_{\epsilon}$. Using the KOMA classes, it should be relatively easy to obtain the look and feel that you like.

Some of the KOMA-Script options are also made available in packages that can easily be loaded in your own favourite class (for example the AMS classes) by including a `\usepackage{}` command with one of the special KOMA-Script packages (see below for details).

Most modern \TeX -distributions already include KOMA-Script; but if your installation does not, then download a copy from CTAN://macros/latex/contrib/koma-script or from <http://developer.berlios.de/project/koma-script3/>. Installation instructions are rather straightforward and covered in detail in the included README and INSTALL files.

2 Page layout using `typearea.sty`

2.1 Some theoretical considerations

When looking at a page of printed text, one may notice that it consists of the actual text and the margins. Sometimes there are headers and footers as well in between the main text body and the top and bottom margin respectively. When the printed text is a book, there was some additional white space that was printed but is hidden in the binding. Setting up a page with optimal margins and text area is called *page layout*.

Page layout doesn't depend only on fashions that change over time and from one culture to another; it also have a firm basis in human visual perception. Some texts are easier to read than others, and the two most important factors determining readability are page layout and the choice of typeface (Bringhurst, 2004).

KOMA-Script implements a series of modern European page layouts using its `typearea` package. The basic tenets of this series are as follows:

- (a) the size of the top margin compared to the bottom margin should be as 1:2;
- (b) for a single-sided document, the left and right margins should have equal width;
- (c) for a double-sided document, the total "inner" margin (i.e., the right-hand margin of the left page plus the left-hand margin of the right page) should equal the width of the "outer" margins (an example of this is shown in figure 5 on the fifth page, page 46, of the document by Willi Egger at http://www.ntg.nl/maps/pdf/30_13.pdf).

Please note that, when talking about page, we refer to the *visible* page, which may or may not be equal to the sheet of paper used to print the page on. In a book, for example, the sheet of paper contains both the page and the binding area.

Knowing how to set the margins in relation to each other is of course not enough. The text width and height should also be determined. The optimal line width is determined by many factors, including:

- typeface and font size used
- interline spacing

- language of the text and average word length
- number of columns
- page dimensions

Larger fonts allow wider lines than smaller fonts; serifed fonts guide the eye along the text and hence allow for wider lines than sans-serifed fonts. More space between lines makes it easier to skip from the end of one line to the beginning of the next and, hence, allow for wider lines; but bear in mind that there is an optimum somewhere and increasing line space beyond this optimum will hinder rather than help the eye. Average word length and language play a role too. Generally, a text in a western language, typeset with a standard serifed font of size 10 to 12 and the standard L^AT_EX line space should contain no more than, on average, fifteen words per line. This usually translates to 60 or 70 characters per line, including spaces and punctuation. Finally, the available space on the page plays an important role. One can imagine that the optimal line width for a paperback novel and a glossy magazine or even a newspaper would be rather different.

The optimal text height is influenced more by our sense of aesthetics than by readability. Nowadays the optimal text height is considered to best be linked to the text width.

When we take all these considerations together, we end up with an algorithm for dividing a page into equally sized rectangles. Suppose we assume nine rectangles horizontally, and nine vertically for a single-sided document (see again figure 5 on the fifth page, page 46, at http://www.ntg.nl/maps/pdf/30_13.pdf). This is called a *division factor* of nine. Mentally shift the gray area of the figure half a rectangle to the right; we would then have a left and right margin of one and a half rectangle each, a top margin of one rectangle, and a bottom margin of two rectangles. Our typearea would then be six rectangles wide and six rectangles high. For the two-page spread of a double sided document all heights would remain the same, but the vertical margins would change. The inner margin would consist of one rectangle, and the outer margin of two rectangles (as shown in the referenced figure). The text width would thus still be six rectangles, so changing from single-sided to double-sided does not change line breaks.

We can use any factor other than nine, as long as it is at least four. The reason for the lower limit of four is that, using the margin size rules described

above, typearea would be one rectangle high and one rectangle wide (which would already probably look silly); a smaller division factor would leave no space for the typearea.

The ideal division factor depends on the page size, the chosen font, etc. For an A4-sized page width a 12pt text as described above, one would probably take a division factor of 15. There are, however ways of having L^AT_EX take care of determining an optimal division factor.

2.2 Implementing the theory using typearea.sty

KOMA-Script implements two options to control the page layout according to the theory described above. This is done using the package typearea.sty, which can be loaded with any document class. The KOMA-Script classes load this package automatically, so when using a KOMA document class, these options can be given in the optional part of the \documentclass[]{} command. The options are called BCOR (Binding CORrection) and DIV (DIVision factor).

Figure 1 shows an example using the standard L^AT_EX_{2 ϵ} report class. We use A4-paper to print a double sided report with the Bookman fonts, yielding an optimal division factor of 15, and we need a binding correction of 12mm because our report will be bound with glue. The first page of the resulting output in figure 2

There are some additional ways of using the DIV option. One of them is using calc rather than an explicit DIV-factor. This will tell L^AT_EX to take care of determining an optimal DIV-factor. This makes life a lot easier for those users who lack the typographical (or mathematical) background to determine the optimum line width themselves. Please see figure 3 for an adaptation of our previous example to include this useful option.

A complete list of ways to use the DIV-option is:

DIV<factor> Set the division factor to be equal to the number <factor>.

```
\documentclass[a4paper,twoside]{report}
\usepackage{bookman}
\usepackage[DIV15,BCOR12mm]{typearea}
\usepackage{lipsum}
\begin{document}
\lipsum[1-8]
\end{document}
```

Figure 1: The source for a basic example of the use of typearea.sty

DIVcalc Calculate the optimal division factor for the current page and font settings.

DIVclassic Calculate a division factor for the current page and font settings that approximates the mediaeval page layout as closely as possible.

DIVcurrent Re-calculate the page layout using the DIV-factor that is currently in use.

DIVdefault Calculate the page layout with the standard value for the current page and font settings, or — lacking a standard value — use *DIVcalc*.

DIVlast Repeat the calculations for the page layout using the same DIV-argument that was used previously.

These uses of the DIV-option seem to imply that it is possible to do the calculations of DIV-factor and page layout multiple times in the same document. This is indeed the case, but we can of course only include the `typearea.sty` package once. If we want to do the necessary calculations again later, we use the command `\typearea[BCOR]{DIV}`. For example, if we want to use *DIVclassic* with an 8mm binding correction, we can use the command `\typearea[8mm]{classic}`. (Note that, as opposed to the options given when including `typearea.sty` with the `\usepackage` command, we do *not* include the letters DIV or BCOR explicitly in argument of the `\typearea` command.)

The option *current* as used in conjunction with DIV is also available for BCOR, for re-calculating the page layout including the existing BCOR value. The other DIV-options are *not* available with BCOR.

These options may seem rather esoteric at first sight, so it is probably best to give an example of their use. See the source in figures 4 and 5. Here we use

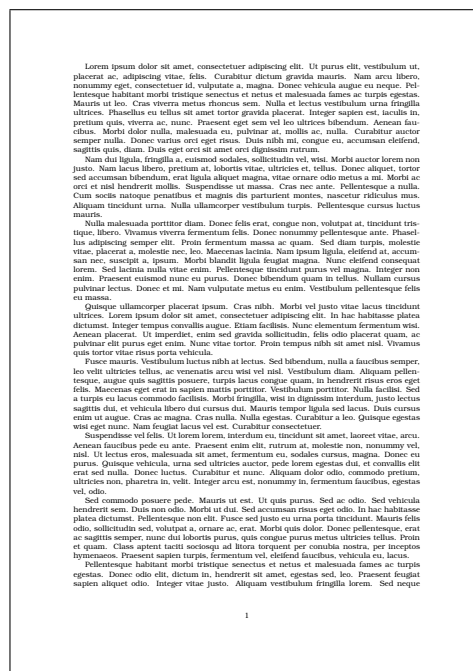


Figure 2: The first page of the output for a basic example of the use of `typearea.sty`.

the KOMA-Script replacement for the report class, which is called `scrreprt`. We also use the default Computer Modern or Latin Modern font, but we want to increase the line spacing to 1.5, using the package `setspace`.

You may notice that the default paper size for KOMA-Script is already A4, and hence this does not need to be included with the `\documentclass` options. Paper size can be changed using any of `letterpaper`, `legalpaper`, `executivepaper`, `aXpaper`, `bXpaper`, `cXpaper`, `dXpaper` and `landscape`. Please note that the `X` in, for example, `aXpaper` is any reasonable integer. So it is possible to make a small booklet on paper one-fourth the size of A4-paper using the `a6paper`-declaration.

You may also notice that the resulting output is markedly different from the output for our first example as presented in figure 2. These differences are caused by the change in line spacing and the change in font. It is probably a good idea to do some experimenting yourself in order to get a feeling for the way the KOMA-Script page layout mechanism works.

Of course we have only been looking at very basic page layout options so far, and we have ignored the fact that for most documents it may not be all that clear what the actual margin is. This may seem a trivial issue. It isn't. Consider headers and footers. Do they belong to the actual text, or to the margin, or perhaps a bit to both? Things get even more complex when we start including marginal notes (the \LaTeX_ϵ command `\marginpar{}` establishes these notes).

Should one of these elements be empty, then obviously it ought to be included with the margin. Similarly, a footer that contains no more than a page number is optically almost empty and should thus be considered part of the margin. If,

```
\documentclass[a4paper,twoside]{report}
\usepackage{bookman}
\usepackage[DIVcalc,BCOR12mm]{typearea}
\usepackage{lipsum}
\begin{document}
\lipsum[1-8]
\end{document}
```

Figure 3: The adapted source for our basic example, now using the `DIVcalc` option.

```
\documentclass[10pt,twoside,BCOR12mm,DIVcalc]{scrreprt}
\usepackage{setspace}
\onehalfspacing
\typearea[current]{last}
\usepackage{lipsum}
\begin{document}
\lipsum[1-8]
\end{document}
```

Figure 4: The source for an example demonstrating the use of `DIVlast` and `BCORcurrent`.

however, the header is full of text (perhaps even underlined), then it is better to include it with the text body rather than with the margin. KOMA-Script is not able to decide whether headers, footers or marginal notes should be considered part of the text body or part of the margin. By default it considers these textual elements to be part of the margins, but this behaviour can be changed using the following options to the `typearea` command:

```
headinclude
headexclude
footinclude
footexclude
mpininclude
mpexclude
```

Using the *include* version of any of these options makes the respective textual element a part of the body; the *exclude* version makes it part of the margin. The size of the page header is 1.25 times the height of a regular line, but this can be changed using the `headlines` option with the `typearea` or `usepackage` command. For example, the command

```
\usepackage[1.8headlines]{typearea}
```

sets the header to a height of 1.8 line height.

It is also possible to set the type area to a particular fixed size and have the `typearea.sty` package calculate the right DIV-value to achieve this. For this use the command

```
\areaset[BCOR]{textwidth}{textheight}.
```

Should you, for example, desire to typeset a book of poetry with square text bodies of 15cm by 15cm and a binding correction of 1cm, use the command

```
\areaset[1cm]{15cm}{15cm}.
```

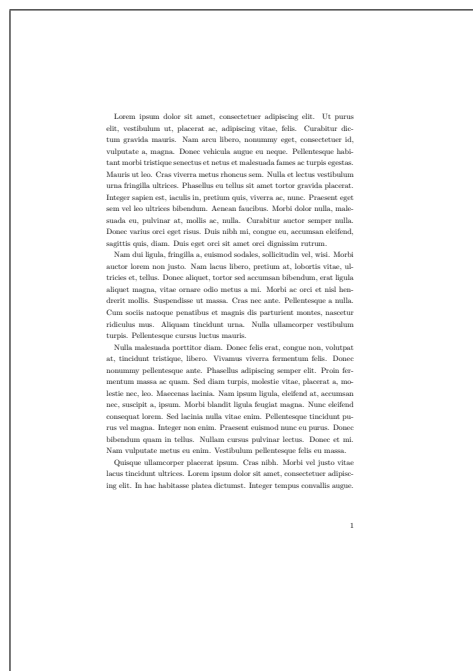


Figure 5: First page of the result for the example of figure 4.

This will give you a text body of the desired size and adjust the margins accordingly.

Should you require particular margin sizes that do not fit into the system described above, then it is better to use the `geometry`-package (Umeki, 2000).

3 A short remark on headers and footers

KOMA-Script has a style file named `scrpage2.sty` that allows all sorts of manipulations, both simple and complex, of headers and footers. It easily surpasses the capabilities of the better known `fancyhdr.sty`. Since this is a large and complicated style file with many options, we think it is better to just make the reader aware of its existence here and refer to the included documentation for now. We intend to write another paper dealing exclusively with the `scrpage2.sty` package for a future issue of *The PracT_EX Journal*.

4 An application: writing letters

This section is intended to show the power of using KOMA-Script. We could have chosen an article, report or book, but we have decided to show a letter, using the `sclttr2` class. We chose a letter for several reasons. First of all, the standard L^AT_EX₂_ε letter class is not used all that often since it looks rather outdated and does not lend itself well to customisation. The `sclttr2` class is quite the opposite: it looks quite modern already, and it is rather easy to make significant customisations and modifications. Also

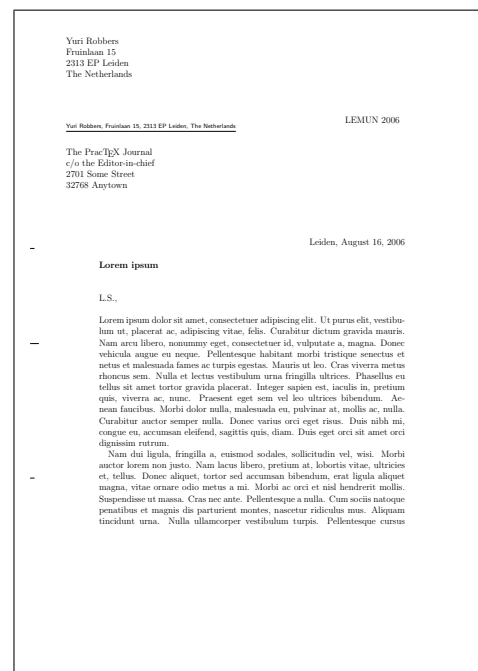


Figure 6: The first page of the output for our very basic KOMA-Script letter from figure 8.

letters tend to be rather short by nature, so it easy to include a complete example in this paper. Finally, while there are quite a few replacements for the other standard classes, there are very few available replacements for the letter class.

We do not intend to show the full power of the `scr1ttr2.sty` package—merely to show some of its options and the effects these options have. This means we will not show all the commands for changing the look and feel of the letter, nor any of the advanced options such as mail-merge set of commands for easily creating standard letters that get sent with slight alternations sent to many people based on databases of recipients. We will show (without explanation) how to save various sets of settings in small data files so that they can be easily reused when sending out, for example, a personal letter (using your home address and perhaps a closing with your first name) or a business letter (using your work address, the company logo, and a digitised signature). For these options see the KOMA-Script documentation (and perhaps we'll write another paper in a future issue of *The PracT_EX Journal*).

Please refer to figure 8 for the example source code of a basic letter (the output is shown in figures 6 and 7).

The `DIV`-option is set to 9, the fontsize to 12, and a few more options are set. The `enlargefirstpage` option is specifically intended for letter writing. Generally on multi-page letters the first page has different settings from the next pages, and adding the option `enlargefirstpage` enables those.

Next a new variable, `company`, is defined in addition to the many pre-defined variables. We use this variable to set a company name in our letter.

We include some packages for including the signature and the dummy text and set a whole bunch of `komavar` variables. These variables can be defined in the letter itself (as shown in this example), but it is generally advisable to create

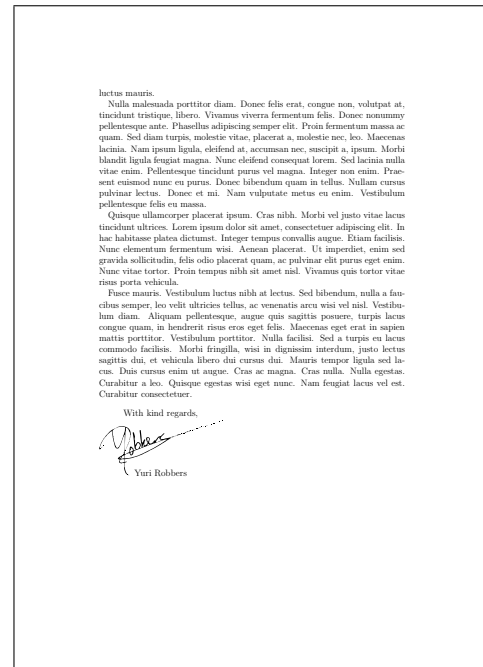


Figure 7: The second page of the output for our very basic KOMA-Script letter from figure 8.


```

\documentclass[%
  DIV=9,
  fontsize=12pt,
  pagesize,
  twoside,
  enlargefirstpage]{scrllttr2}

\newkomavar{company}
\setkomavar{location}{\ifkomavareempty{company}}{\%
  \usekomavar*{company}%
  \usekomavar{company}}\}
}

\setkomavar{place}{Leiden}
\setkomavar{company}{LEMUN 2006}

\usepackage{graphicx}
\usepackage{epsf}
\usepackage{lipsum}

\nexthead{}

\setkomavar{fromlogo}{%
  \includegraphics[width=2cm]{LEMUN-logo2006}}
\setkomavar{fromname}{Yuri Robbers}
\setkomavar{fromaddress}{Fruinlaan 15\2313 EP Leiden\The Netherlands}
\setkomavar{fromphone}{+31-71-5121365}
\setkomavar{fromfax}{+31-71-5144543}
\setkomavar{fromemail}{robbers@lemun.org}
\setkomavar{fromurl}{http://www.lemun.org}
\setkomavar{frombank}{Postbank 9307157}
\pagestyle{empty}
\begin{document}
  \begin{letter}{%
    The Prac\TeX\ Journal\c/o the Editor-in-chief\2701 Some Street\32768 Anytown}
    \setkomavar{subject}{Lorem ipsum}
    \opening{L.S.,}

\lipsum[1-5]

    \closing{With kind regards,\}
\includegraphics[height=25mm]{yuri}\*[-20mm]
  }
  \end{letter}
\end{document}

```

Figure 8: The source code for a basic KOMA-Script letter. Note that the `setkomavar` and `newkomavar` statements could all be moved to a separate `.lco` file for easy reuse.

```

\documentclass[%
  asymTypB,
  DIV=9,
  fontsize=12pt,
  pagesize,
  twoside,
  enlargefirstpage]{scr1ttr2}

\setkomavar{place}{Leiden}
\setkomavar{company}{LEMUN 2006}% new variable defined in asymTypB.lco

\usepackage{graphicx}
\usepackage{epsf}
\usepackage[english]{babel}
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{lucimatx}
\usepackage{lipsum}

\nexthead{}

\setkomavar{fromlogo}{\includegraphics[width=2cm]{LEMUN-logo2006}}
\setkomavar{fromname}{Yuri Robbers}
\setkomavar{fromaddress}{Fruinlaan 15\\2313 EP Leiden\\The Netherlands}
\setkomavar{fromphone}{+31-71-5121365}
\setkomavar{fromfax}{+31-71-5144543}
\setkomavar{fromemail}{robbers@lemun.org}
\setkomavar{fromurl}{http://www.lemun.org}
\setkomavar{frombank}{Postbank 9307157}
\pagestyle{empty}
\begin{document}
  \begin{letter}{%
    The Prac\TeX\ Journal\\c/o the Editor-in-chief\\2701 Some Street\\32768 Anytown}
    \setkomavar{subject}{Lorem ipsum}
    \opening{L.S.,}

\lipsum[1-5]

    \closing{With kind regards,\\%*[1cm]
\includegraphics[height=25mm]{yuri}\\%*[-20mm]
}
    \end{letter}%
\end{document}

```

Figure 9: The source code for a fancier KOMA-Script letter. Note that a .lco file named asymTypB.lco is included in the documentclass-command.

a so-called .lco file, a *letter class option* file. Any definitions made in this .lco file will be automatically loaded by adding its name (minus the .lco-extension) to the \documentclass options.

Then we begin the document and the letter (with addressee) and create an opening and a subject, put the main body (in this case using dummy text generated with \lipsum), and a closing, including a picture file with a signature. Then we close letter and document.

Note that the resulting letter is rather basic but does include fold marks, and has the addressee in the exactly the right spot for an ISO-compliant window envelope. There is even an underlined return address in tiny letters in the window.

Even without further explanation, you will probably be able to adapt this simple example for your own use. But let's see some more of the power of KOMA-Script and create a slightly fancier letter. As a matter of fact, let's go all out and use one of the fanciest pre-defined styles that can be downloaded as a .lco-file from <http://www.komascript.de>. The source of our fancier letter, including some commands to include the beautiful Lucida Bright fonts from Bigelow & Holmes, is shown in figure 9, and the source code of asymTypB.lco in the appendix (page 14). The two output pages are displayed in figures 10 and 11.

Most users will probably not be willing or able to create such complex .lco files as shown in the appendix; however, several of them are freely available at <http://www.komascript.de>, and simpler ones can easily be created even by beginning users using the KOMA-Script documentation. Help is available at the forum at <http://www.komascript.de>.

This fancy example letter creates a column with relevant information about the sender, including an optional logo, in the right-hand margin of the first page.

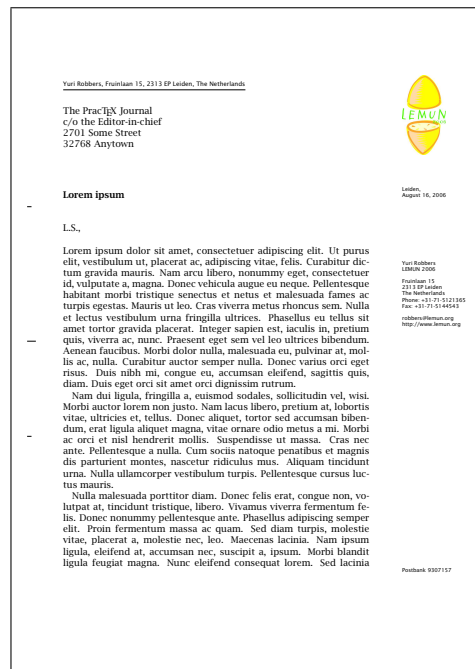


Figure 10: The first page of the output of our fancier KOMA-Script letter from figure 9.

The addressee and an underlined return address in small print are again displayed at exactly the right place for sending the letter in an ISO-compliant window envelope. Note that, since a .lco file was created anyway, the definition of the company keyword was moved there too. This .lco can be reused for business letters of various styles. It is straightforward to include multiple .lco-files in one document.

5 Concluding remarks

We hope a little bit of the power and configurability of KOMA-Script has been demonstrated to the reader. Of course the scope of this paper is necessarily limited; much more has been explained in the book by one of the authors (Kohm & Morawski, 2005).

KOMA-Script offers a uniform interface to document layout, with a replacement class for each of the original L^AT_EX₂ ϵ -classes, except the slides class. The standard layout is well-thought out, modern-looking and based on European rather than American typographical ideas. KOMA-Script is highly configurable and adaptable. Several extension packages for standard L^AT_EX₂ ϵ are no longer necessary when using the KOMA-script classes. Should one, however, decide to use another class, KOMA-Script still has much to offer through its classes: typearea.sty, scrpage.sty, scrtime.sty (has not been mentioned before but is used for displaying system time), scrdate.sty (used for displaying system time and date respectively in L^AT_EX-documents), verb+scraddr.sty+ (used for handling address files), and scrfile.sty (adds additional hooks pre and post inputting files and loading classes or packages). These classes offer advanced ways of manipulating document layout using powerful options and commands that are easy to learn.

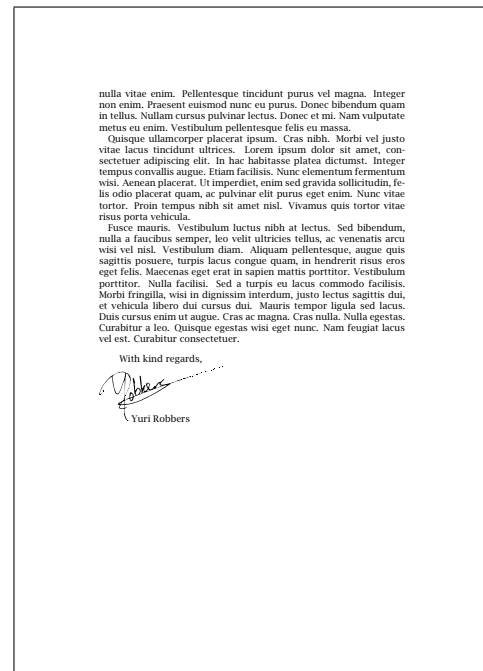


Figure 11: The second page of the output of our fancier KOMA-Script letter from figure 9.

We urge readers to give KOMA-Script a try and to experiment with the many available options.

We thank David Walden and the anonymous referees for their helpful comments.

References

BRINGHURST, R. (2004). *The Elements of Typographic Style, 2nd Edition*. Vancouver: Hartley & Marks.

KOHN, M. & MORAWSKI, J.-U. (2005). *KOMA-Script. Eine Sammlung von Klassen und Paketen für L^AT_EX₂ ϵ* . Heidelberg: DANTE.

UMEKI, H. (2000). The *geometry* package. [CTAN://macros/latex/contrib/geometry/](http://ctan://macros/latex/contrib/geometry/).

A The contents of the asymTypB.lco-file.

```
\ProvidesFile{asymTypB.lco}%
      [2005/04/09 v0.2a unsupported LCO-file]%
\LoadLetterOption{DINmtext}%
\setlength{\oddsidemargin}{\useplength{toaddrhpos}}%
\addtolength{\oddsidemargin}{-1in}%
% Take care that the shift stays intact even after recalculating the page
% layout (see Kohm & Morawski 2005, section C.7)
\l@addto@macro{\@typearea@end}{%
  \setlength{\oddsidemargin}{\useplength{toaddrhpos}}%
  \addtolength{\oddsidemargin}{-1in}%
}
\ifx\AtBeginDocument\@notprerr
  \KOMAOptions{DIV=last}%
\else
  \AtBeginDocument{%
    \g@addto@macro{\@typearea@end}{%
```

```

        \setlength{\oddsidemargin}{\useplength{toaddrhpos}}%
        \addtolength{\oddsidemargin}{-1in}%
    }%
}%
\fi
% continue as per description
\@setlength{firstheadvpos}{0pt}%
\@setlength{firstheadwidth}{\paperwidth}%
\@setlength{firstfootvpos}{\paperheight}%
\@addtoplength[-]{firstfootvpos}{\useplength{toaddrvpos}}%
\@addtoplength{refvpos}{-1.5\baselineskip}%
\newkomavar{company}%
\@newplength{infocolwidth}%
% Kohm & Morawski 2005, C.7. Modifikationen (Modifications)
\ifdim \textwidth<0.666\paperwidth
    \@setlength{infocolwidth}{.22222\paperwidth}%
\else
    \@setlength{infocolwidth}{0.1667\paperwidth}%
\fi
% continue as per description
\firsthead{%
    \fontsize{7}{8}\sffamily
    \hspace*{\fill}%
    \begin{picture}(0,0)%
        \put(0,0){\parbox[t]{\useplength{infocolwidth}}{%
            \vspace{\useplength{toaddrvpos}}%
            \usekomavar{fromlogo}%
        }}%
    }%
    \put(0,0){\parbox[t]{\useplength{infocolwidth}}{%
        \raggedright
        \vspace{\useplength{refvpos}}%
        \vspace{\useplength{refaftervskip}}%
        \usekomavar{place}\usekomavar{placeseparator}\\
        \usekomavar{date}\\[10\baselineskip]
    }}

```

```

\usekomavar{fromname}
\ifkomavareempty{company}{}{%
  \\\
  \usekomavar{company}%
}\[\baselineskip]
\usekomavar{fromaddress}\\\
\usekomavar*{fromphone}\usekomavar{fromphone}\\\
\usekomavar*{fromfax}\usekomavar{fromfax}%
\[\baselineskip]
\usekomavar{fromemail}\\\
\usekomavar{fromurl}\\\
}%
}%
\put(0,0){\parbox[t]{\useplength{infocolwidth}}{%
  \raggedright
  \vspace{\useplength{firstfootvpos}}%
  \vspace{-\footskip}%
  \usekomavar{frombank}\\\%
}%
}%
\end{picture}%
\hspace*{\useplength{infocolwidth}}%
}%
\l@addto@macro\@firstheadfootfield{\setkomavar{date}{}}
\nexthead{%
  \begingroup
  \@tempswatrue
  \if@twoside\ifodd\number\value{page}
    \else\@tempswafalse\fi\fi
  \if@tempswa
    \makebox[0pt][l]{%
      \hspace*{-\oddsidemargin}\hspace{-1in}%
      \hspace{\paperwidth}%
      \hspace{-\useplength{infocolwidth}}%
      \normalfont\fontsize{7}{8}\sffamily
    }
  \fi
}

```

```

\begin{picture}(0,0)
  \put(0,0){\parbox[t]{\useplength{infocolwidth}}{%
    \vspace*{-\topmargin}\vspace{-1in}%
    \vspace{-\headheight}%
    \vspace{\useplength{toaddrvpos}}%
    \begin{tabular}{@{}c@{}}
      \usekomavar{fromlogo}\\
      \usekomavar{company}\\
    \end{tabular}%
  }}%
}%
\end{picture}%
}%
\fi
\endgroup
}%
\pagestyle{headings}%
\endinput

```


[Journal](#)
[home page](#)
[General](#)
[information](#)
[Submit an](#)
[item](#)
[Download](#)
[style files](#)
[Copyright](#)
[Contact us](#)

THE PracTeX Journal



The memoir class

Peter Wilson

Abstract

The memoir class is essentially the book and report classes with lots of bells and whistles as it includes the functions of over 30 popularly used packages. It can also simulate the appearance of article-class documents and provides a basis for producing the typewritten-like manuscripts which some publishers ask for.

Peter Wilson (PhD) first started using LaTeX soon after it became available for creating camera-ready copy for ISO International Standards, plus internal company reports, and conference and journal submissions, and one book. He has written two LaTeX classes, something over 20 packages, and two series of fonts using Metafont. His is a past Editor-in-Chief of IEEE Computer Graphics and Applications (which accepted LaTeX manuscripts). He says, "Having worked long and hard in the UK and US in industry, government and academia I am now retired." He can be reached at `herries.press@earthlink.net`

- [PDF version of paper](#)
- [Comment on this paper](#)
- [Send submission idea to editor](#)

Page generated August 19, 2006 ; [TUG home page](#); [search](#); [contact webmaster](#).

The memoir class

Peter Wilson

Email `herries.press@earthlink.net`

Abstract The memoir class is essentially the book and report classes with lots of bells and whistles as it includes the functions of over 30 popularly used packages. It can also simulate the appearance of article class documents and provides a basis for producing the typewritten-like manuscripts which some publishers ask for.

1 Introduction

For nearly 20 years I was involved in using L^AT_EX to produce camera-ready copy of International Standards, in particular ISO 10303 (STEP). As the standard grew — it now consists of thousands of pages spread across many publications — and ISO and various committees kept changing their minds about what they wanted, I grew increasingly frustrated with having to keep modifying the internals of the class and packages that I had developed. Separately I became interested in book design and felt that there was a need for a class that would support layout experiments. I worked on this in a desultory fashion for several years and eventually produced something that I felt might be generally useful.

The memoir class was first released in 2001 and since then has proven to be reasonably popular. The class can be used as a replacement for the book and report classes, by default generating documents virtually indistinguishable from ones produced by those classes. The class includes some options to produce documents with other appearances; for example an article class look or one that looks as though the document was produced on a typewriter with a single font, double spacing, no hyphenation, and so on. In the following I use the term ‘standard classes’ to denote the book and report classes and, when appropriate, the article class as well.

The memoir class includes the functionality of many packages, for instance the `tocloft` package for controlling the table of contents or methods similar to the

fancyhdr package for designing your own headers. The built-in package functions are mainly related to document design and layout; memoir does not touch upon areas like those that are covered by the babel or hyperref packages or any related to typesetting mathematics. On the other hand it is easy to configure a work produced with memoir to meet a university's thesis layout requirements.

memoir has improved substantially since it was first released — over 50 L^AT_EXers have provided code or suggestions for improvements. The class is included in the TUG T_EX distributions and the latest version of the class and its supporting documentation is always available from CTAN:macros/latex/contrib/memoir.

2 General considerations

The class is a large one consisting of about 10,000 lines of L^AT_EX code documented in a 400 page report; there is no need for most users to look at this. There is a separate comprehensive User Manual which runs to about 300 pages and from time to time an Addendum is released noting extensions to the class. The first part of the Manual discusses some aspects of book design and typography in general, something that I haven't come across in the usual L^AT_EX books and manuals. This is intended to provide a little background for when you design your own printed documents.

The standard classes provide point options of 10, 11, or 12 points for the main body font. memoir extends this by also providing 9, 14, and 17 point options. The width of the text block is automatically adjusted according to the selected point size to try and keep within generally accepted typographical limits for line lengths; you can override this if you wish. The class also provides easy methods for specifying the page layout parameters such as the margins, both side margins and those at the top and bottom of the page; the methods are similar to those of the geometry package.

The page layout facilities also include methods, like those provided by the fancyhdr package, for defining your own header and footer styles, and you can have as many different ones as you wish. In fact the class provides seven styles to choose from before having to create your own if none of the built-in styles suit you. The styles are all displayed in the Manual.

Sometimes it is useful, or even required, to place trimming marks on each page showing the desired size of the final page with respect to the sheet of paper

that is used in the printer. This is provided by the `showtrims` option. A variety of trim marks are provided and you can define your own if you need some other kind.

2.1 Sectioning styles

Handles are provided for designing and using your own styles for chapter titles and such. The class comes with six predefined chapter styles ranging from the default look to a style that mimics that used in the *Companion* series of L^AT_EX books. There is even one which uses words instead of numerals for chapter numbers. The Manual shows at least one example of each style and over 20 are shown in Lars Madsen's collection at <http://www.imf.au.dk/system/latex/artikler/MemoirChapStyles>.

For those who like putting quotations near chapter titles the `epigraph` environment can be used.

The options for changing `\section` and lower level titles are more constrained, but generally speaking document design, unless for advertisements, should be constrained.

Sometimes, but particularly in novels, a sectional division is indicated by just leaving a blank line or two between a pair of paragraphs, or there might be some decorative item like three or four asterisks. Commands are available for typesetting such anonymous divisions.

In the standard classes the sectioning commands have an optional argument which can be used to put a short version of the section title into the table of contents and the page header. `memoir` extends this with a second optional argument so you can specify one short version for the contents and an even shorter one for page headers where space is at a premium.

2.2 Captions

`memoir` incorporates the code from my `ccaption` package which lets you easily modify the appearance of figure and table captions; bilingual captions are available if required, as are captions placed at the side of a figure or table. This also supports subfigures and subtables along the lines of the `subfig` package, plus letting you define your own new kinds of floats together with the corresponding 'List of ...'.

3 Tables

Code from the `array`, `dcolumn`, `delarray` and `tabularx` packages is integrated within the class. To improve the appearance of rules in tabular material the `booktabs` package is also included.

Multipage tabulations are often set with the `longtable` or `xtab` packages, which can of course be used with the class. For simple tabulations that may continue from one page to the next, `memoir` offers a ‘continuous tabular’ environment. This doesn’t have all the flexibility provided by the packages but can often serve instead of using them.

More interestingly, but more limited, the class provides ‘automatic tabulars’. For these you provide a list of simple entries, like a set of names, and a number of columns and the entries are automatically put into the appropriate column. You choose whether the entries should be added row-by-row or column-by-column.

4 Verse

The standard classes provide a very simple *verse* environment for typesetting poetry. This is greatly extended in `memoir`. For example in the standard classes the verse stanzas are at a fixed indentation from the left margin whereas `memoir` lets you control the amount of indentation so that you can make a poem appear optically centered within the textwidth.

Stanzas may be numbered, as can individual lines within a poem. There is a special environment for stanzas where lines are alternately indented. Also you can define an indentation pattern for stanzas when this is not regular as, for example, in a limerick where the 3rd and 4th of the five lines are indented with respect to the other three. It is not always possible to fit a line into the available space and you can specify the particular indentation to be used when a ‘logical’ verse line spills over the available textwidth, thus forming two or more typeset ‘physical’ lines.

5 End matter

Normally appendices come after the main body of a book. The class provides some methods for introducing appendices at the end, or you can place one or more appendices at the end of selected chapters if that suits you better.

memoir also lets you have more than one index and an index can be set in either the normal double column style or as a single column which would be more appropriate, say, for an index of first lines in a book of poetry. The titles of any bibliography or indexes are added to the table of contents, but you can prevent this if you wish.

The class provides a set of tools for making glossaries or lists of symbols, the appearance of which can, of course, be easily altered. The `makeindex` program is used to sort the entries. An example is shown in the current version of the Addendum. The most recent addition to the class provides configurable end notes which can be used as well as, or instead of, footnotes.

6 Miscellaneous

As already noted, the Manual for memoir runs to some 300 pages and it is impossible to cover everything in a short article. Suffice it to say that hooks and macros are provided for most aspects of document layout; for instance, footnotes can be as normal, typeset in two or three columns, or all run into a single paragraph. You can create new verbatim-like environments, read and write information in external files, design your own style of `\maketitle`, convert numbers to words, reserve space at the bottom of a page, and so on and so forth.

A Packages

Most packages work with the memoir class. Apart from from those indicated below the main exception is the `hyperref` package. This package modifies many of the internals of the standard classes but does not cater for all of the differences between memoir and the standard ones. If you wish to use `hyperref` with memoir then you must use the `memhfixc` package¹ after using `hyperref`. For example like:

1. `memhfixc` is supplied as part of the memoir distribution.

```

\documentclass[...]{memoir}
...
\usepackage[...]{hyperref}
\usepackage{memhfixc}
...
\begin{document}

```

The memoir class includes code either equivalent to, or extensions of, the following packages; that is, the set of commands and environments is at least the same as those in the packages: abstract, appendix, array, booktabs, ccaption, chngcntr, chngpage, crop, dcolumn, delarray, enumerate, epigraph, framed, ifmtarg, ifpdf, index, makeidx, moreverb, needspace, newfile, nextpage, patchcmd, shortvrb, showidx, tabularx, titleref, titling, tocbibind, tocloft, verbatim, verse. The class automatically ignores any `\usepackage` or `\RequirePackage` related to these.

The class also provides functions similar to those provided by the following packages, although the commands are different: fancyhdr, geometry, sidecap, subfigure, titlesec. You can use these packages if you wish, or just use the capabilities of the memoir class.

[Journal](#)
[home page](#)
[General](#)
[information](#)
[Submit an](#)
[item](#)
[Download](#)
[style files](#)
[Copyright](#)
[Contact us](#)

THE PracTeX Journal



LaTeX curricula vitae with the CurVe class

Didier Verna

Abstract

CurVe is a LATEX2e class for writing curricula vitae (cv). It provides a set of commands to create headers, rubrics, entries in these rubrics etc. CurVe will then format your cv with a consistent layout while you can just concentrate on the contents. The layout of a CurVe cv is highly customizable. CurVe also has a very special feature known as the flavor mechanism: it is able to manage different "flavors" (versions) of your cv simultaneously. CurVe is distributed under the terms of the LPPL license. This paper describes the features available in version 1.11.

Mr. Verna is a computer scientist working for the prestigious EPITA Research and Development Laboratory in the south of Paris, France. He is married and a father of two children, and he is a semi-professional jazz guitarist and singer. In addition to his research and teaching, Mr. Verna has published the book "GNU autoconf, automake, libtool" (New Ryders) and is a software developer (of, for example, XEmacs, Gnus and BBDB) and has created several add-on packages for LaTeX, such as FiXme, FiNK, QCM and CurVe. Mr. Verna can be reached at `didier@lrde.epita.fr`

- [PDF version of paper](#)
- [Comment on this paper](#)
- [Send submission idea to editor](#)

Page generated August 19, 2006 ; [TUG home page](#); [search](#); [contact webmaster](#).

L^AT_EX curricula vitae with the *CurV_e* class

Didier Verna

Email didier@lrde.epita.fr

Website <http://www.lrde.epita.fr/~didier>

Address EPITA / LRDE, 14-16 rue Voltaire, 94276 Le Kremlin-Bicêtre, France

Hobby Also a semi-professional Jazz guitarist and singer

Abstract *CurV_e* is a L^AT_EX2_ε class for writing curricula vitae (cv). It provides a set of commands to create headers, rubrics, entries in these rubrics *etc.* *CurV_e* will then format your cv with a consistent layout while you can just concentrate on the contents. The layout of a *CurV_e* cv is highly customizable. *CurV_e* also has a very special feature known as the *flavor mechanism*: it is able to manage different “flavors” (versions) of your cv simultaneously. *CurV_e* is distributed under the terms of the LPPL license. This paper describes the features available in version 1.11.

1 Getting *CurV_e*

CurV_e can be obtained from any CTAN archive, at [CTAN:macros/latex/contrib/curve](http://ctan.org/macros/latex/contrib/curve). You can also download it directly from my website, at the URL above. Please follow the links on the left menu.

If you are a Debian unstable user (unstable referring to Debian, not you), unofficial source and i386 packages are available. The package name is *curve*. Here’s the `source.list` entry to use:

```
deb http://www.lrde.epita.fr/debian/ unstable/i386/  
deb-src http://www.lrde.epita.fr/debian/ sid/source/
```

For installation instructions, please read the README file included in the distribution.

2 First Contact

Figure 1 shows the output of *CurVe* for a minimal cv with absolutely no customization of any kind. Only the basic information for a cv is provided.

2.1 The document layout

Note the different parts of a cv made by *CurVe*, and their default layout: as you can see, a cv begins with a small photo, two headers (upper left and upper right) in which you usually put your name, address, email, whether you're married and so on. After these headers come a title and a subtitle.

The remainder of the document is composed of sections called “rubrics” in the *CurVe* terminology . A rubric represents a major topic that you want to detail in your cv. Typical rubrics are “Professional Experience”, “Education” and the like. Rubrics have a title, centered on the page by default, and appear under the form of properly aligned “entries” (see below). If a rubric has to be split across different pages, its title will be repeated automatically.

An entry is an item of information related to the rubric under which it appears. An entry has a “contents”, and an optional “key” under which it is classified. For instance, under the “Education” rubric, you could state that you got a Ph.D. in computer science in the year 2000. In that case, the year would be the entry’s key, and the “Ph.D. in computer science” part would be the entry’s contents. *CurVe* aligns both keys and contents together. Keys are optional in order for you to classify several entries together (without repeating the same key over and over again). There is also an option that will make *CurVe* automatically skip identical keys in consecutive entries.

Additionally, you might want to further split your rubrics into “subrubrics”.

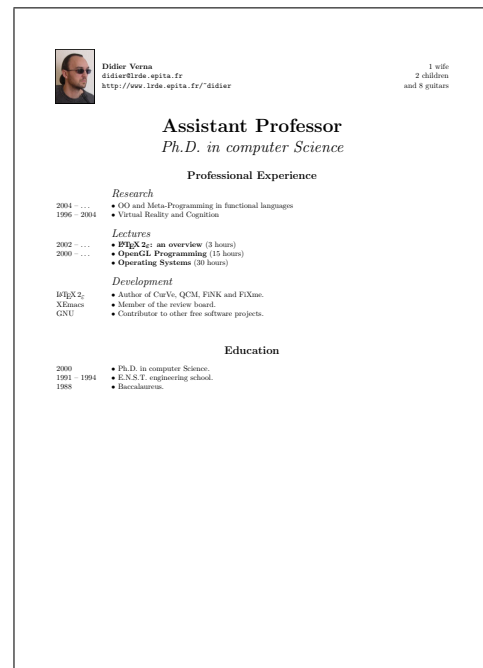


Figure 1: A minimal cv

For instance, in the example above, the “Professional Experience” rubric is further split into three subrubrics: “Research”, “Lectures”, and “Development”. Subrubrics are displayed in alignment with the entries’ contents by default, but are formatted differently so that they remain distinguishable.

Finally, note the presence of a small bullet in front of each entry’s contents. This is called a “prefix” in the *CurVe* terminology. This prefix is a visual clue that helps in distinguishing one entry from another, especially in the case of multiline entries, or entries sharing the same key.

2.2 The code

2.2.1 The main source file

Let us now examine the source code for this cv. The main file is given in figure 2. The only tiny bit of customization here is the use of the geometry package to provide a very wide text area on the page, something very common for a cv. *CurVe* itself does not do anything to modify page dimensions, margins, etc. There are very powerful packages to deal with this so there is no point in duplicating the functionality.

Otherwise, the rest is pretty straightforward: you provide headers with the `\leftheader` and `\rightheader` commands, the `\photo` command takes an image file name (sans extension if your `graphicx` package is properly configured), and the title and subtitle the usual way. Without any surprise, headers and titles are formatted by calling `\makeheaders` and `\maketitle` at the top of your document.

```
\NeedsTeXFormat{LaTeX2e}
\documentclass[a4paper]{curve}

\usepackage
[nohead,nofoot,hmargin=1.5cm,vmargin=1.5cm]
{geometry}

\leftheader{\textbf{Didier Verna}}\
\texttt{didier@lrde.epita.fr}\
\texttt{%
  http://www.lrde.epita.fr/~{didier}}
\rightheader{1 wife\
  2 children\
  and 8 guitars}
\photo{didier}

\title{Assistant Professor}
\subtitle{Ph.D. in computer Science}

\begin{document}

\makeheaders
\maketitle

\makerubric{experience}
\makerubric{education}

\end{document}
```

Figure 2: A minimal source file

One surprise, however, is that the content of the rubric is not available directly in the main source file: each rubric must remain in its own file. For instance, to include an “education” rubric, you must write it to a file named `education.tex` and include it, as shown in the example, with the `\makerubric` command.

2.2.2 The rubric source files

The code associated with the “Professional Experience” rubric is shown in figure 3. As you can see, the whole contents of a rubric file must be enclosed in a rubric environment. This environment takes one mandatory argument which specifies the rubric’s title. When a rubric crosses several pages, its title is restated with a “continuation” text appended.

You create entries in your rubrics by calling the `\entry*` macro. This macro behaves much like an `\item` within a list environment: there is one optional argument that defines the entry’s key (otherwise empty), and the entry’s contents are simply anything that follows, up to the next entry or subrubric. The presence of a star in the command’s name is for historical reasons: there is also a `\entry` command that takes the whole entry’s contents as a mandatory argument (hence within braces).

In order to create a subrubric, you simply call the `\subrubric` command. Its mandatory argument specifies the subrubric’s title.

```
\begin{rubric}{Professional Experience}

  \subrubric{Research}
  \entry*[2004 — \ldots]
    OO and Meta-Programming in functional
    languages
  \entry*[1996 — 2004]
    Virtual Reality and Cognition

  \subrubric{Lectures}
  \entry*[2002 — \ldots]
    \textbf{\LaTeXe: an overview} (3 hours)
  \entry*[2000 — \ldots]
    \textbf{OpenGL Programming} (15 hours)
  \entry*
    \textbf{Operating Systems} (30 hours)

  \subrubric{Development}
  \entry*[\LaTeXe]
    Author of CurVe, QCM, FiNK and FiXme.
  \entry*[XEmacs]
    Member of the review board.
  \entry*[GNU]
    Contributor to other free software
    projects.

\end{rubric}
```

Figure 3: A rubric source file

3 Customizing the appearance

In the previous section, we saw that writing a cv with *CuV*e is a fairly easy process. Once you are satisfied with the contents, you might still want to tweak the layout to your personal taste (or constraints). Since the layout generated by *CuV*e is extremely customizable, it would be out of the scope of this paper to describe every possible configuration option (besides, *CuV*e is fully documented, and the documentation is, I hope, well written). Rather, let us take back our minimal example and tweak it a bit to illustrate some of the possible customizations.

Figure 4 shows the result of the customization process. The full source code for this version is given in Appendix A. Note that there is no modification of the rubrics files whatsoever.

You can see that the photo is now on the right. This is achieved by passing an optional argument to `\photo` like this: `\photo[r]{didier}`. You can use `l` (the default), `c` or `r` meaning that the photo will appear on the left, center, or right.

Additionally, the headers are now aligned on top, rather than vertically centered. This is achieved by passing an optional argument to `\makeheaders` like this: `\makeheaders[t]`. You can use `t` (for top), `b` (for bottom) or `c` (for center; the default).

Also, note the use of different fonts for the titles, rubrics, keys *etc.* In *CuV*e virtually any text category comes with a command to modify the font used for it. For instance, to change the appearance of the rubrics' titles, use the `\rubricfont` command, to change that of the title, use the `\titlefont` command and so on.

You will also notice that we changed the prefix for something fancier. The command to do that is `\prefix`, to which we passed `\ding{52}` (from the `pifont` package) in that particular case. If you prefer, you can get rid of the prefix altogether.

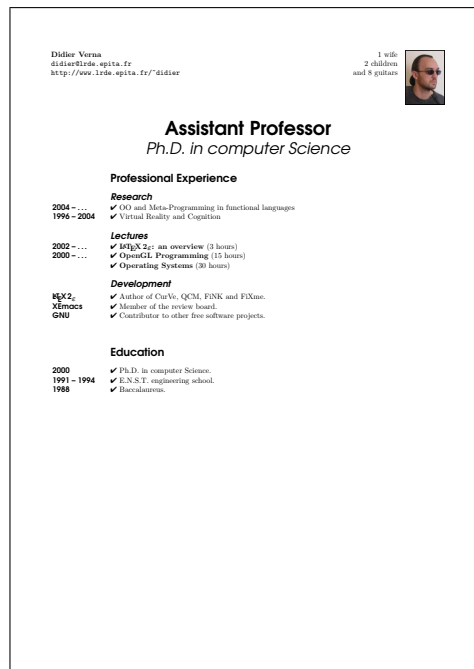


Figure 4: A customized cv

Finally, note how we changed the alignment of the rubrics' titles: they are not centered anymore, but rather aligned with the entries' contents (and also the subrubrics for that matter). The command to do that is `\rubricalignment`. There are up to 6 possible alignments for the rubrics' titles. Horizontal alignment of the subrubrics and the keys can also be changed.

4 Advanced usage

In this section, we describe two advanced features of *C_{ur}V_e* that might come in handy.

4.1 Bibliography

Most scientists include their own list of publications in their cv. The first thing you can do is create your own bibliography manually, just as you would create a rubric.

If you find this process too cumbersome, however, you can use the standard L^AT_EX bibliography mechanisms with *C_{ur}V_e*: the `thebibliography` environment is supported along with its `\bibitem` companion. The behavior is actually that of a rubric environment with its `\entry*` companion (with an empty prefix however). Hence, the layout of the bibliography remains consistent with the rest of the cv. This fact has two implications however: firstly, the argument to the `thebibliography` environment is unused in *C_{ur}V_e* (but remains for compatibility with the rest of L^AT_EX) because *C_{ur}V_e* itself formats the keys and contents properly aligned. Secondly, the bibliographic environment **must** reside in its own file, as any other rubric. Don't forget that if you happen to write the environment manually.

If you want to use B^IB^TE_X, that's also possible of course. Do it as you would do in any other document. You will probably issue a `\nocite{*}` command followed by a call to `\bibliography`. In *C_{ur}V_e*, the `bbl` file is actually used as if it is just another rubric on the cv.

Finally, note that *C_{ur}V_e* is compatible with the `bibentry` package.

4.2 Flavors

It is often desirable to maintain several slightly divergent versions of one's cv at the same time. For instance, when I was looking for a job some time ago, I had a version of my cv emphasizing Artificial Intelligence, and another emphasizing Distributed Virtual Reality. Only the title and some entries in the "Professional Experience" rubric were a bit different; the main skeleton basically remained the same.

CurV_e provides an easy-to-use mechanism for maintaining different "flavors" of your cv at the same time. You basically write different versions of (some of) your rubrics in different files, tell *CurV_e* which flavor you want to format (*CurV_e* can even ask you which one to use directly) and that's it. *CurV_e* will use the global skeleton, and whenever it finds a rubric file specialized for that particular flavor, it will use it. Otherwise, it will simply fall back to the default one (no particular flavor).

The flavor mechanism works by assigning a pre-extension to rubric file names. For instance, suppose you want to make a special flavor of your cv emphasizing "distributed virtual reality". You would call this flavor "dvr", and write the modified "Professional Experience" rubric into a file named `experience.dvr.tex`.

In order to tell *CurV_e* which flavor to use, you then call the `\flavor` macro like this: `\flavor{dvr}`. Instead of using the `\flavor` macro, you can make *CurV_e* ask you at run-time which flavor to use by passing the `ask` option to it.

In order to implement the flavor mechanism, the L^AT_EX macro `\input` is redefined to look for "flavored" files first. This is actually very nice because you can use it if you want to make different flavors of text that does not belong in rubrics. For instance, suppose you want a special version of the subtitle of your cv for the flavor dvr: just create a file called `subtitle.dvr.tex` and put something like `\subtitle{special subtitle}` in it. Do something similar for the default subtitle. Now go to the skeleton of your cv, and write `\input{subtitle}` in the preamble. That's it. You have different subtitles in your different cv flavors.

5 Conclusion

In this paper, we have barely scratched the surface of the features *CurV_e* has to offer, but provided enough, I hope, to make the reader feel like giving it a try. For

instance, there are many more customization options available (control of spacing, most notably). Please refer to the official documentation for more information.

To conclude, let me just mention some other important features of $\mathcal{C}_u\mathcal{V}_e$: it supports the standard L^AT_EX options (font size, paper geometry *etc.*) and page styles. It comes with built-in support for AUC-T_EX (command completion in Emacs or XEmacs *etc.*). Thanks to many contributors, 9 foreign languages are also supported (and customizable).

$\mathcal{C}_u\mathcal{V}_e$ has some caveats though, mostly as a consequence of the underlying implementation based on the L^AT_EXtable package. In order to help, the official documentation now provides a FAQ and a section with some hints on what you can and can't do within a "longtable".

The next big step for $\mathcal{C}_u\mathcal{V}_e$ (which will trigger an increment of the major version number) will be the support for the notion of "themes". The idea is to make it easy to use predefined layouts, and also to support official cv styles, like the european one. This has been on my todo list ... for some time now!

A Customized main source file

```
\NeedsTeXFormat{LaTeX2e}
\documentclass[a4paper]{curve}

\usepackage[nohead,nofoot,hmargin=1.5cm,vmargin=1.5cm]{geometry}

\leftheader{\textbf{Didier Verna}\\
\texttt{didier@lrde.epita.fr}\\
\texttt{%
http://www.lrde.epita.fr/~{}didier}}
\righthead{1 wife\\
2 children\\
and 8 guitars}
\photo[r]{didier}

\titlefont{\fontfamily{pag}\selectfont\bfseries\Huge}
\title{Assistant Professor}

\subtitlefont{\fontfamily{pag}\selectfont\itshape\huge}
\subtitle{Ph.D. in computer Science}

\rubricfont{\fontfamily{pag}\selectfont\bfseries\Large}
\rubricalignment{cl}

\subrubricfont{\fontfamily{pag}\selectfont\bfseries\itshape\large}

\keyfont{\fontfamily{pag}\selectfont\bfseries}

\usepackage{pifont}
\prefix{\ding{52}}

\begin{document}

\makeheaders[t]
\maketitle

\makerubric{experience}
\makerubric{education}

\end{document}
```

[Journal](#)
[home page](#)
[General](#)
[information](#)
[Submit an](#)
[item](#)
[Download](#)
[style files](#)
[Copyright](#)
[Contact us](#)

THE PracTeX Journal



ConTeXt Starters

Joe Hogg

Abstract

This article presents two beginning projects using ConTeXt. The first project is a letterhead that can be used for business and personal correspondence mailed in an envelope with an address pane. The second project is a four-page brochure done by the Botany Committee for Ape Awareness Day last November at the Los Angeles Zoo and Botanical Gardens.

Joe Hogg is a volunteer docent at the Los Angeles Zoo and Botanical Gardens. He can be reached by email at Joseph.Hogg@bigfoot.com

- [PDF version of paper](#)
- [ConTeXt source file for letterhead](#)
- [Example of resulting letterhead](#)
- [ConTeXt source file for four-page brochure](#)
- [Example of resulting four-age brochure](#)
- [Comment on this paper](#)
- [Send submission idea to editor](#)

Page generated August 19, 2006 ; [TUG home page](#); [search](#); [contact webmaster](#).

CON_TE_XT Starters

Joe Hogg

Abstract This article presents two beginning projects using CON_TE_XT. The first project is a letterhead that can be used for business and personal correspondence mailed in an envelope with an address pane. The second project is a four-page brochure done by the Botany Committee for Ape Awareness Day last November at the Los Angeles Zoo and Botanical Gardens.

Joe Hogg is a volunteer docent at the Los Angeles Zoo and Botanical Gardens. He can be reached by email at Joseph.Hogg@bigfoot.com.

At the 2005 PracT_EX conference in Chapel Hill, I saw CON_TE_XT demonstrated on several projects and wanted to try it. The exposure to different ways to typeset documents was one of the benefits of my attending this conference. At the time, I didn't view CON_TE_XT as a replacement for L^AT_EX and I still don't. CON_TE_XT, L^AT_EX and Plain T_EX can all be used for their separate strengths.

1 Letterhead

My first project using CON_TE_XT was letterhead that I could use for business and personal correspondence. I like to use envelopes with address panes and wanted to include the recipient's address in the letter in a position that would show through the pane in the envelope.

The complete source file, `letterConTeXt.tex` and associated PDF file accompany this article.

You may have to adjust the spacing for font size and address length in order to be sure the recipient's address shows through the pane in the envelope. If you don't use this type of envelope, simply comment out or remove the address commands.

My correspondents have liked the appearance of my letters. However, the first time they receive my letter in US#10 business envelopes with the address showing through, they think I'm sending an invoice!

2 Four-page Brochure

In the late fall, the Los Angeles Zoo and Botanical Gardens hosts a conservation event called Ape Awareness Day. Various handouts and exhibits are included with information about apes and their environment. This four-page brochure lists seven prominent plants in areas where apes live. The text was written by the Chairman of the Botany Committee and illustrated with photos from various Internet sites. I did the layout and typesetting with `CONTEXT` with the goal of placing the photos to the right of the text.

I started with letter-size paper, but the content was too long for two pages. Legal-size paper worked and allowed for a brochure layout with a single piece of paper folded in the middle. Processing the document produced a four-page PDF file. Using this file, I imposed the brochure with Acro Software's *cutePDF* and printed it on a color laser printer, single sheet of paper printed both sides, then folded. I usually score the paper along the fold in order to make a sharp crease and a tight fold.

The complete source file, `brochureConTeXt.tex` and associated PDF file accompany this article.

3 Wrap Up

These examples are my first attempt to use `CONTEXT`. Surely the design of the documents and their coding can be improved and I would appreciate any comments you have. Nevertheless, if you can use any of the coding for your projects, you are welcome to do so.

[Journal](#)
[home page](#)
[General](#)
[information](#)
[Submit an](#)
[item](#)
[Download](#)
[style files](#)
[Copyright](#)
[Contact us](#)

THE PracTeX Journal



Creating Pocket-size Books Using LaTeX

D.V.L.K.D.P. Venugopal

Abstract

This article deals with creating pocket-sized books of A7 size using LaTeX in a quick and dirty method.

Mr. Venugopal is a stenographer by profession. He has been using computers since the two floppy PC stage. A former lover of WordPerfect, his first encounter with TeX/LaTeX was in the year 2000. At present exploring various packages, experimenting with Omega, MetaPost and MetaFont and teaching and/or explaining advantages of TeX/LaTeX over wordprocessors are his hobbies. You can contact him at venugopal_duvvuri@rediffmail.com

- [PDF version of paper](#)
- [Example of six A4 pages for telephone directory, prior to folding and cutting](#)
- [Comment on this paper](#)
- [Send submission idea to editor](#)

Page generated August 19, 2006 ; [TUG home page](#); [search](#); [contact webmaster](#).

Creating Pocket-sized Books Using L^AT_EX

D.V.L.K.D.P. Venugopal
Banaras Hindu University
Varanasi, India*

Abstract This article deals with creating pocket-sized books of A7 size using L^AT_EX in a quick and dirty method.

1 The Need

For day-to-day work in an office or for personal use, many people like to keep telephone numbers and other contact information handy. For this purpose, at my institution we generally keep a small telephone diary in which to write names and numbers. Since we carry this in our pockets, there is danger of it getting torn or soiled preventing us from reading the numbers we wrote in it; and, if we lose it, then we have to start from square one. In addition, our university has a large Electronic Private Automatic Branch Exchange or EPABX system. For these reasons, it is desirable to have an institution-wide printed version of a pocket-sized booklet of telephone numbers.

2 Pre-L^AT_EX Era

Prior to 1995, we were wholly dependent on the telephone directory printed by our university press. There were fewer telephones then, so managing the information was easy. But from 1995 onwards there was an information technology revolution and the telephone, which was considered a luxury until then, became a necessity. Frequent change of telephone numbers due to establishment of new telephone exchanges was a feature of this period. As our office is the nerve center

*Correspondence: D.V.L.K.D.P. Venugopal, Personal Assistant, Vice-Chancellor's Office, Banaras Hindu University, Varanasi 221 005, India. E-mail : venugopal_duvvuri@rediffmail.com

of the university and we have to deal with many officials inside and outside of the university, it was natural for our office to want more frequent updates of the telephone book, and we started creating them ourselves.

Computers were introduced in our office in the year 1995. So we started compiling a telephone directory on a computer. As the word processor we used didn't have much capability, we originally prepared the telephone directory by photocopying two A4 (210×297 mm) printed pages on to one A4 page by reduction. We prepared only one or two copies for our exclusive use.

Later versions of the word processor had a feature for printing booklets, and our work became somewhat less tiresome. Still, creating directories smaller than the size of an A5 page (148×210 mm) was a problem.

3 Tools Available in L^AT_EX

About four years after I began learning L^AT_EX, I had gained enough confidence to convert the telephone directory into L^AT_EX. Though converting the data to L^AT_EX was easy, creating a pocket-sized telephone directory without cutting and pasting was a challenge.

On a cursory glance through the ConT_EXt manual [1], I found that one can typeset beautiful books using ConT_EXt. As it was developed by a publishing company, it has all the facilities for creating quality books. But as I had never used it, I continued my search.

While studying the ProT_EXt Package Manager, I discovered the leaflet, twoup, booklet and pdfpages packages. I first explored leaflet [2]. This package is useful for creating leaflets of conferences, symposia, etc., that contain six pages on an A4 page. In the leaflet documentation I found that it was inspired by the booklet and twoup packages. The twoup package uses the dvips route and requires the user to be familiar with Postscript files and their printing [3]. This twoup package is dependent on 2up, a generic T_EX package. The booklet package gives us the facility to create booklets of size A5 [4]. However, I wanted to create a booklet of size A7 (74×105 mm, or about 3×4.125 inches).

Then I stumbled upon the pdfpages package [5]. It has many tricks up its sleeve. Using this package one can combine many small PDF documents into a single document (useful for keeping separate publications together), create a new PDF document from selected pages of other PDF documents (useful to keep ab-

stract pages of many related articles together), or to insert diagrams from another PDF file.

4 Creating the pocket-sized booklet

For readers who are not familiar with the A series of page sizes, two A5 pages side-by-side in a portrait orientation fit on one A4 page in a landscape orientation, two A6 pages fit on one A5 page, and two A7 pages fit on an A6 page. Thus, eight A7 pages fit on one A4 page, and 16 A7 pages fit two sides of one A4 sheet of paper.¹

Our first step in creating the A7 sized directory we desired was to create a telephone directory of A5 size, which came out to be 48 pages long. The telephone directory earlier created using the word processor was saved as a text file and edited appropriately for use with L^AT_EX. We used the `article` class and `longtable` package. The directory so created was named `pocket-td.pdf`.

4.1 Second Step

We then created another file, `pocket.tex`, consisting of the following lines of L^AT_EX, which processed the file `pocket-td.pdf`:

```
\documentclass[a4paper]{article}

\usepackage[final]{pdfpages}

\begin{document}
\includepdf[pages=-,nup=1x2,landscape,signature=48]{pocket-td.pdf}
\end{document}
```

L^AT_EXing `pocket.tex` placed pairs of our A5 pages on each A4 page in the resulting `pocket.pdf` file, creating 24 A4 pages from our initial 48 A5 pages.

1. For a full description of paper sizes see http://en.wikipedia.org/wiki/A4_Paper_Size.

4.2 Third step

Then we created another file named `pocket2.tex` with the following lines of \LaTeX :

```
\documentclass[a4paper]{article}

\usepackage[final]{pdfpages}

\begin{document}
\includepdf[pages=-,nup=1x2,landscape,signature=24]{pocket.pdf}
\end{document}
```

Running `pocket2.tex` through \LaTeX to process the file `pocket.pdf` produced a new file, `pocket2.pdf`, containing 12 A4 pages, each of which had four A6 size pages on it.

4.3 Final Step

Another file, named `pocket3.tex`, was created containing the following lines of \LaTeX :

```
\documentclass[a4paper]{article}

\usepackage[final]{pdfpages}

\begin{document}
\includepdf[pages=-,nup=1x2,landscape,signature=12]{pocket2.pdf}
\end{document}
```

On \LaTeX ing the above file, we obtained the file `pocket3.pdf` containing the six A4 pages of a booklet of A7 size. (See the link to the file on the HTML page for this paper.)

Print these pages yourself on both sides of three A4- (or letter-) size pages, fold the pile of three pages at the center and cut along the fold to get A5 pages. Fold the pile of A5 pages at the center and cut along the fold to get A6 pages. Fold the pile of A6 pages in the center, staple pages at the fold, and, voilá, you get the telephone directory of A7 size.

4.4 Simplifying the process

In the above process we have seen that we have to compile four documents. But we may have to change or add numbers frequently in the telephone directory. Once we created the above files there is no need to follow all the steps. Instead we created a `pocket.bat` file (we are using MS Windows). The entries in the above batch file are as follows:

```
cd..  
cd teldirtex  
pdflatex pocket-td  
pdflatex pocket  
pdflatex pocket2  
pdflatex pocket3
```

So whenever we make corrections in the main file, we save it and open the MS-DOS prompt and type `pocket`. All the files are \LaTeX ed, simplifying a tedious process.

5 Conclusion

The above process is a quick and dirty method. \TeX perts will be able to create better macros and an easier process. Our process is best suited for pocket-sized books containing up to 32 pages, and 48 pages is OK. If the pocket book contains more than 48 pages, we would have to follow another route.

Acknowledgments

I thank the anonymous reviewers and Dave Walden, guest editor of this issue, for their help in refining the presentation of this paper.

Biography of the author

Mr. Venugopal is a secretarial assistant by profession. He has been using computers since the two-floppy PC stage. A former lover of WordPerfect, his first

encounter with T_EX/L^AT_EX was in the year 2000. At present his hobbies are exploring various T_EX packages, experimenting with Omega, MetaPost and MetaFont, and teaching and/or explaining advantages of T_EX/L^AT_EX over word processors.

References

- [1] Hagen, H. ConT_EXt – The Manual. 2001. www.pragma-ade.com.
- [2] Niepraschk, R., Schmidt, W. and Gäßlein, H. The document class leaflet. 2004.
- [3] Hansen, M.L. The twoup Package. 2005.
- [4] Wilson, P. Printing booklets with L^AT_EX. 2005.
- [5] Matthias, A. The pdfpages Package. 2004.

[Journal](#)
[home page](#)
[General](#)
[information](#)
[Submit an](#)
[item](#)
[Download](#)
[style files](#)
[Copyright](#)
[Contact us](#)

THE PracTeX Journal



Travels in TeX Land: Experiences Refining Page Layout for a Book

David Walden

Abstract

In this column in each issue I muse on my wanderings around the TeX world. In my last column (www.tug.org/pracjournal/2006-2/walden) I described some methods I use to make it easier to draft large documentation projects (i.e., books). In this column I describe my experience with taking final steps of turning a book manuscript into a published book.

David Walden is retired after a career as an engineer, engineering manager, and general manager involved with research and development of computer and other high tech systems. You can contact him via the email address you can find by scrolling to the bottom of the page at www.walden-family.com; alternatively send email to his attention via pracjournal@tug.org.

- [PDF version of paper](#)
- [Comment on this paper](#)
- [Send submission idea to editor](#)

Page generated August 19, 2006 ; [TUG home page](#); [search](#); [contact webmaster](#).

Travels in T_EX Land: Final Layout of a Book

David Walden

Abstract In this column in each issue I muse on my wanderings around the T_EX world. In my last column (www.tug.org/pracjourn/2006-2/walden) I described some methods I use to make it easier to draft large documentation projects (i.e., books). In this column I describe my experience of taking the final steps of turning a book manuscript into a published book.

1 Caveat

I worry slightly about the suitability of my following descriptions: maybe people more more expert with L^AT_EX than I would recommend better approaches and “deprecate” the approaches I have figured out. My concern is reduced, however, by the thought that there is a lot to be said for doing whatever is expedient to get the job done — in this case getting a book to the printer — and being motivated by the experience to seek better approaches next time. Whatever the quality of my practices, it may be useful for other non-expert L^AT_EX users and first book publishers to see the long list of issues that came up for me.

These experiences also will make me consider seriously using something other than L^AT_EX’s book class for my next book. Perhaps I’ll try the Memoir class or ConT_EXt.

2 Waiting for copy editing

My manuscript was more or less finished, and I sent it to the copy editor for her review. While she edited the manuscript, I began to tune up the manuscript in anticipation of final layout.

2.1 Fonts

I had drafted the manuscript using Palatino for my roman font, Helvetica for sans serif, and Courier for the typewriter font. Steve Peter suggested that I might switch to Minion as the roman font, and helped me do so. (I had asked Steve, a professional book designer and

publisher, to be my consultant in the use of \LaTeX in the design of my book as this is the first time I have taken a book length document all the way to publication in \LaTeX . I have “gone all the way” with \LaTeX for journal publications, and I have developed book length manuscripts in \LaTeX . But with my previous \LaTeX -developed books, the publisher in both cases retypeset the book using QuarkXpress based on my conversions of my \LaTeX files to plain text via HTML — see, for instance, section 4 of www.tug.org/TUGboat/Articles/tb24-2/tb77walden.pdf.)

I tried downloading and installing a set of files that claimed to be Minion Pro for \TeX , but I couldn’t get it to work. Eventually, Steve provided me with a set of Minion files he had created for \TeX , and he talked me through installing them. I already had Minion Pro on my computer as part of Photoshop CS2 and Illustrator CS2, so I felt comfortable that I was licensed to have and use Minion. I continued to use Helvetica for sans serif because I only used it in a few \TeX -based tables and I had many tables drawn with Illustrator that used Helvetica and wanted the lettering in all tables to be in the same font. I switched to using what came with Minion for roman fonts. Thus, my personal class file was changed from

```
\RequirePackage{mathpazo}
\RequirePackage[scaled=.95]{helvet}
\RequirePackage{courier}
```

to

```
\RequirePackage{minion}
\RequirePackage[scaled=.95]{helvet}
\RequirePackage{courier}
```

This column uses Minion.

However, I didn’t like the way Courier (only used for URLs) looked with Minion, so I dropped the Courier line above

```
\RequirePackage{minion}
\RequirePackage[scaled=.95]{helvet}
```

and fell back on the \TeX default Computer Modern typewriter font, which I think looks OK with Minion (and which I also use in this column).

Having changed to Minion, I had to make some adjustments. The first adjustment was to change the Roman numerals of my part numbers to small caps; in the table of contents and running head, full caps Roman numerals looked too big compared with old style

numerals. Next, I had to change all instances of `\$` (\$) to `\textdollaroldstyle{}` (\$) which was defined in one of the Minion files from Steve Peter. (My book is about business management, so quite a few dollar signs were present.)

This was my first experience with a font that used old style figures. (Old style figures are number characters where 1, 2, and 0 are the size of letters such as a, m, and x; the numbers 3, 4, 5, 7, and 9 hang down like g, p and y; and 6 and 8 stick up like b, h, and t. Old style figure are also known as text figures, hanging figures, and lower case figures; the other kind of figures where the number characters are all the size of capital letters are known as titling, ranging, and lining figures. For a discussion of the two types of figures, see page 46 of the 2005 paperback edition of Robert Bringhurst’s *The Elements of Typographic Style*, Hartley and Marks, Point Roberts, WA.) I quickly became used to seeing old style figures; however, I had the characters “10×” several places in one of my book chapters, as in “a 10× change in the market” which Andrew Grove talks about in his book *Only the Paranoid Survive* (paperback edition, New York: Currency, 1999). This looked OK with lining (capital letter size) numbers and a capital letter X in Palatino or plain text (“10X change”), but with the conversion to Minion, I had to define

```
\newcommand{\TenX}{\{10\kern -.01in$\texttimes$}}
```

to find something that I thought looked good, as in the quote above from Grove.

With the switch to Minion, I also switched to 12pt for my main text:

```
\documentclass[12pt]{btbook}
```

The style specified in the above command is my own style for this book as described in my previous column. I chose 12 point primarily because my co-author’s and my eyes are not getting any younger, and 12 point with its default leading (interline spacing) looks nice and airy to my not-too-discerning eye. (I am also using 12 point for this column.) However, using `normalsize` text, the \LaTeX default, for block quotes doesn’t make them distinctive enough to my eye, despite the greater indentation on the left and right. Using `\small` for block quotes with its default sizing looked too small to me. Thus, I redefined `\small` in my class file using a pattern that Steve Peter showed me:

```
\renewcommand\small{%
  \@setfontsize\small{10.7pt}{13.05pt}
  \abovedisplayskip 9.5\p@ \@plus2.5\p@ \@minus5\p@
  \abovedisplayshortskip \z@ \@plus\p@
  \belowdisplayshortskip 5.25\p@ \@plus2.75\p@ \@minus2.5\p@
  \belowdisplayskip \abovedisplayskip}
```

Initially I didn't know what much of the above means (although, Peter Flynn's presentation at the PracTeXo6 conference, which presumably will be published in an upcoming issue of this journal and/or *TUGboat*, explained what it means), but it produced a pleasing result for block quotes and the few other places where I used small text — in the bibliography, end notes, and so forth. I redefined L^AT_EX's block quote macros to use my version of `\small`:

```
\renewenvironment{quote}
  {\list{}{\rightmargin\leftmargin}%
   \small%added by me
   \item\relax}
  {\endlist}

\renewenvironment{quotation}
  {\list{}{\listparindent 1.5em%
   \small%added by me
   \itemindent \listparindent
   \rightmargin \leftmargin
   \parsep \z@ \@plus\p@}%
   \item\relax}
  {\endlist}
```

2.2 Right justification

The justification of the right margins of my draft manuscript didn't look as straight as they are with another book I recently had published that the publisher typeset with Quark-Express. A few months before (while interviewing Frank Mittelbach for TUG — see www.tug.org/interviews), I learned that so-called microtype capabilities exist with pdfT_EX. Consequently, I decided to try

```
\RequirePackage{microtype}
```

I printed out the documentation for the microtype package, but I didn't read it as microtype just worked without any additional commands (luckily, Minion was one of the five or six fonts for which configuration files came with the microtype package). The right margins immediately looked straighter, there were less line overflows, and fewer lines were hyphenated.

Microtyping apparently involves two primary capabilities. First, hyphens, periods, etc., are allow to extend *slightly* past what should be the right margin, and this makes the right

margin *look* more straight. Second, justifying lines using the inter-word spacing, the actual characters in a line (and inter-letter spacing?) are allowed to shrink or expand *slightly* to better fit the line, in addition to the normal expansion and shrinkage of interword spaces. Microtyping is turned on for this column.

2.3 Figure placement

The macros for figure and table formatting that I defined in my column in the previous issue (www.tug.org/pracjourn/2006-2/walden) did not include a parameter for specifying page position. I could see that I was going to want to be able to better force figures to the top or bottom of pages, or sometimes to the middle of a page with text above and below. I immediately converted the table macro and all calls of it to have an additional argument (there were only a few L^AT_EX-produced tables in the whole book). I had *many* figures, however, so for them I defined a new macro that included the additional positioning argument and began to switch to its use a figure at a time, as I came across combinations of text and a figure where L^AT_EX's default figure placement was not adequate. (Will Robertson's paper in this issue deprecates the use of figure macros such as I discussed in my last column and mention again here. My justification for using them is that they help *me* handle my *large* number of figures.)

2.4 Bibliography

I actually made the decision I am about to discuss before I sent my manuscript to the copy editor; I made it because I was using a copy editor. For my last book developed in L^AT_EX I used Bib_T_EX. Bib_T_EX was very convenient for experimenting with different formatting and citation systems for the bibliography. However, in this case of using a copy editor and with a modest-size bibliography, I decided that it would be easier to make the copy editor's edits in my computer files if I just typed the approximate format for each entry in the correct order (alphabetical by author and chronological for each author), and then edited these bibliography entries according to the copy editor's red pen marks on the hard copy of the manuscript.

Therefore, I wrote a little macro to do gross formatting of bibliography entries, which follows along with an example of its use (defining a macro named `\ref` below was a **serious mistake** — see the note at the end of section 3.3):

```

\def\ref#1#2{\vskip 4pt
  \vbox{\noindent\small
    \hangindent = 1pc \textbf{#1.}\hskip.05in#2}}

\ref{Ackoff81}{Ackoff, Russell L. \textit{Creating the Corporate
Future: Plan or Be Planned For}. New York: Wiley, 1981.}

\ref{Bower97}{Bower, Joseph L. Harvard Business School Case:
‘‘Teradyne: The Aurora project’’ (revised March 29, 1999). Boston:
Harvard Business School Press, 1997.}

```

This produces the following:

Ackoff81. Ackoff, Russell L. *Creating the Corporate Future: Plan or Be Planned For*. New York: Wiley, 1981.

Bower97. Bower, Joseph L. Harvard Business School Case: “Teradyne: The Aurora project” (revised March 29, 1999). Boston: Harvard Business School Press, 1997.

In the main text I cited such bibliography entries as follows

Russell Ackoff talks of type 1 system properties and type 2 (or “emergent”) system properties (Ackoff81).

Obviously, this is a somewhat nonstandard citation method, but I like it and I can use it since I am self-publishing.

I treated the bibliography as just another unnumbered backmatter chapter rather than using the typical command to insert the bibliography.

This approach to the bibliography in this case is indicative of my tendency to do a tiny bit of programming and a good bit of manual editing rather than taking time to look around for and (especially) learn to use capabilities that may already exist.

2.5 Integrating the copy editor’s work

At this point groups of chapters began to come back from the copy editor, each page covered with red ink marks and writing indicating necessary or suggested changes to the manuscript and suggestions for possible improvement of the presentation. (When writing, I try to insist that editors make no changes to my files, until the final layout pass; however, some publishers these days insist that their editors edit my files, flagging the changes they have made for my review. I am not fond of this “modern” practice of editors actually changing

my files. Fortunately, the copy editor for this book was as reactionary in this matter as I am and insisted on editing a double spaced printout of the manuscript with her red pen.) I processed all of the copy editor’s mandatory and suggested changes (each chapter took me a long, e.g., 10-hour, day to do — a couple of weeks altogether); next came about four solid days of proofreading and minor changes (and a couple of hard-to-think-about major changes that I had procrastinated about making from before the manuscript went to the copy editor).

3 Waiting for co-author’s review

I sent the much improved manuscript to my co-author to review (he is a non-native English speaker — we collaborate on the content but I do all the writing in English). While waiting for his review, I made some more changes on my way to a final layout.

3.1 Log file

I searched the PDF output once again for instances of a double question mark indicating places where \LaTeX could not resolve cross-references, and I took a look at the log file for the first time (until now I saw little point in plowing through vast numbers of log-file comments about line overflows, and I normally run pdf \TeX from WinEdt where the \LaTeX window closes after a successful compilation without leaving me anything to scroll through on the screen). I immediately noticed that there were lots of “label multiply defined” messages involving tables, which resulted from my table macro in fact having a redundant `\label` command, and which was trivially fixed. But there was so much other stuff still in the log file that I set it aside and vowed to look again later (I never did).

3.2 Ellipses

My copy editor was never happy with the spacing of ellipses that were done with \TeX ’s `\dots` command. I only used ellipses a few times in my whole manuscript, mostly in block quotes. Therefore, I created much less general versions of ... where I could explicitly control the spacing:

```
%dots for main text
\def\BDS{3pt}
%three dots
```

```

\def\MYdots{\hbox{\hspace{\BDS}.\hspace{\BDS}.\hspace{\BDS}.\hspace{\BDS}}}
%period and three dots = four altogether
\def\fsMYdots{\hbox{.\hspace{\BDS}.\hspace{\BDS}.\hspace{\BDS}.\hspace{\BDS}}}

%dots for block quote text
\def\SDS{2pt}
%three dots without end spaces
\def\MINsmallldots{\hbox{.\hspace{\SDS}.\hspace{\SDS}.}}
%three dots
\def\sMYdots{\hbox{\hspace{\SDS}\MINsmallldots\hspace{\SDS}}}
%period and three dots = four altogether
\def\fsMYdots{\hbox{\hspace{.3pt}.\hspace{\SDS}\MINsmallldots\hspace{\SDS}}}

```

The above macros are intended to be used directly ... as here ... abutting text on both sides, thus inhibiting line breaking at either side of an ellipsis. With so few ellipses in my entire manuscript, my plan was to manually note instances where there should be a line break before or after an ellipsis and manually allow the correct thing to happen. Such manual intervention (rather than trying to develop a redefinition of `\dots` that better suits my needs) has the disadvantage that if I later change the text (for example, in a reuse of the text in another document), the manual fix might no longer be the correct thing. My thought is that I could deal with this eventuality by then redefining all of my special “dots” macros to be equivalent to `\dots` or by simply doing a Replace-All of each instance of one of my dots commands by `\dots`.

3.3 End notes

When I was originally drafting my book, I used footnotes because it is convenient to have them on the same page as the text they refer to. However, everyone I run into in professional publishing seems to disapprove of footnotes because they make the book look too scholarly. Before I sent the manuscript to the copy editor, I converted all the footnotes to end-of-chapter notes with the following definitions:

```

\renewcommand{\footnote}{\endnote} %comment out to not have end notes

\newcommand{\dumpendnotes}
{
\medskip
\begin{group}

```

```
\setlength{\parindent}{0pt}\setlength{\parskip}{1ex}
\theendnotes\endgroup \setcounter{endnote}{0} }
```

I added a `\dumpendnotes` command at the end of each chapter. In this way, my copy editor was able to also edit my notes in a realistic format along with each chapter.

However, publishing professionals also seem not to like end-of-chapter notes — I guess it is considered rude to force readers to look in different places in the book for notes for different chapters. Therefore, I decided to move all the end notes to a single “chapter” near the end of the book. In particular, I happened to be skimming through a copy of *Misquoting Jesus* (Bart D. Ehrman, Harper, 2005) at the same time, and tried to mimic this book’s style for end-of-book chapter notes.

I looked in *The L^AT_EX Companion* (second edition) and on `comp.text.tex` for how to combine all chapter end notes in one place, but neither of these was much help to me (perhaps because I was not smart enough to understand what I was reading). Therefore, I modified the `endnotes.sty` file some more (in my last column I mention how I had already modified it, and renamed it `dw-endnotes.sty` to slightly change the format of end notes). Now, I changed the end of each chapter to only reset the end note counter, i.e.,

```
\setcounter{endnote}{0}
```

and I put a single `\theendnotes` command in a new Notes “chapter” near the end of the book (beginning with `\chapter{Notes}` and appropriate redefinition of the running heads). The result of having only one `\theendnotes` command is that all the endnotes for all chapters go in succession in one `.ent` file with no distinctions among notes from different chapters, although the note numbers restart at 1 at the beginning of the notes from each chapter. My changes to the portion `endnotes.sty` which it typesets the end notes included:

- commented out where it typesets the heading for a Notes section, since we only needed this once — and not with the notes for each chapter
- included a test for whether a note number was equal to 1
- defined a new counter with which to keep track of chapter numbers
- inserted a test for whether a note number being read from the `.ent` file is equal to 1 (i.e., is the first note of a chapter); if it is, the chapter number counter is incremented by one and the word “Chapter” and the chapter number are typeset with appropriate vertical spacing

(If someone wants my actual code for the above, please ask.)

Unfortunately, for some reason my end note processing changes put an erroneous space in instances of inter-end-note cross references, e.g., instances where one end note said, See `note~\ref{fn:whatever}` on page `\pageref{fn:whatever}`, and the label `fn:whatever` was defined in the referenced end note; this printed as

5. See note 8 on page 243.

All non-inter-end-note cross references worked fine; thus, rather than trying to figure out what was going on, I replaced `note~\ref{fn:whatever}` with an absolute note number, e.g., `note~8`, in the two instances where I had of such cross references, and I put a comment in the source file to remind me to remove the absolute references if I later reuse this text in another document.

(When this paper was being edited for publication, Barbara Beeton pointed out the I should not have redefined `\ref` (which \LaTeX already uses for cross-references. Rather, I should have called my macro something else, such as `\Ref`. This also indicates the value of using `\newcommand` for defining macros which would have told me I was redefining an existing macro. Some lessons I seem to have to learn over and over again.)

3.4 Table of contents

Using the \LaTeX defaults and the size font I had specified, the table of contents spilled slightly onto a third page. I wanted the table of contents to fit on two pages, and I wanted the page boundary to be between parts of the book. Again, I looked in the documentation and at the `comp.text.tex` discussion group but didn't find what I needed. So I hunted around in the files `latex.ltx` and `book.cls` (and a few other files) until I found the `\@starttoc` command and the `\l@chapter` command.

I first looked at the `.toc` file resulting from the compilation of my book. That was full of lines of the form

```
\contentsline {chapter}{\numberline {3}Identify Need}{45}
\contentsline {section}{\numberline {3.1}Humanitas and principle 1}{45}
```

I found `\contentsline` defined in `latex.ltx`:

```
\def\contentsline#1{\csname l@#1\endcsname}
```

Although I don't completely understand this definition, it seemed like it was defining a command that had as its name the first argument of the `\contentsline` command. I

hunted around for `l@chapter` and found in `book.cls` the definition which controlled the spacing before chapter lines in the table of contents. Then I looked for where the table of contents is typeset and found the `\@starttoc` command in `latex.ltx`.

I redefined both `\@starttoc` and `\l@chapter` in my class file so there was less vertical space before chapter lines and `\baselineskip` was a little smaller throughout the table of contents. By trial and error with the amounts of vertical space, I was able to get the page break to be where I wanted it to be without the table of contents flowing onto a third page.

Altogether I spend several hours until I had the page breaks of the table of contents where I wanted them.

Then, after talking with the printing company, I changed the text block size of the whole book which put the table of contents page break in the middle of the subsections of a chapter. Remembering a tip from Walter Cazzola in response to a question of mine on `comp.text.tex` six years ago, I inserted

```
\addtocontents{toc}{\raggedbottom\protect\vspace{.5in}}
\chapter{Build New Business} \flushbottom
```

at the beginning of the chapter in question which forced the chapter and chapter subsections lines onto the next page without putting extra vertical space in either page of the table of contents. (Barbara Beeton has told me `\flushbottom` has no effect in the above.)

3.5 Part and chapter headings

Later I decided that \LaTeX 's default chapter heading format was a little oversized compared with my text. I experimented a little bit with `\usepackage{fncychap}` but the package's built-in styles seemed more different than I wanted.

So I found the following definitions in `book.cls` and modified the vertical spacing and the font sizes, as follows:

```
\def\@makechapterhead#1{%
  \vspace*{30\p@}%
  {\parindent \z@ \raggedright \normalfont
    \ifnum \c@secnumdepth >\m@ne
      \if@mainmatter
        \LARGE\bfseries \@chapapp\space \thechapter
        \par\nobreak
        \vskip 15\p@
      \fi
    }
```

```

\fi
\interlinepenalty\@M
\LARGE \bfseries #1\par\nobreak
\vskip 30\p@
}}

\def\@makeschapterhead#1{%
\vspace*{30\p@}%
{\parindent \z@ \raggedright
\normalfont
\interlinepenalty\@M
\LARGE \bfseries #1\par\nobreak
\vskip 30\p@
}}

```

This caused some interaction with the table of contents, and I had to undo the last change I described in the previous section.

3.6 Kerning

I noticed that there was too much space between the *W* and the *a* in my last name (Walden) in \Large size on the title page and in italics at the end of the preface. As a result, I took a many-hour look at how kerning tables are specified in T_EX's font files; but ultimately I gave up trying to modify and recompile these tables and fonts. As an interim measure, I just inserted a kerning command at the few places that were particularly obvious to me, for example:

```
\Large{\scshape David W\kern-2pt{ }alden}
```

4 Preparing a ready-to-print file

While I was doing the work described in the last section, my co-author finished his review and suggested a number of changes to the manuscript. We debated some of his proposals, and I made the changes we agreed upon in my L^AT_EX and .eps image files. We also began to discuss the cover design (which I did as an Illustrator file — see the following page).

Breakthrough Management

Shoji Shiba and
司馬正次 David Walden

Confederation of Indian Industry

Breakthrough Management
Shiba and Walden



Prof. Shiba is my Guru. His teachings that are a part of this book have helped me to transform Sona Koyo into an ambidextrous organization which is necessary to support breakthrough practices. His lessons on leadership have helped me personally. This book is a must for all students of management and young managers who desire to be the leaders of tomorrow.

- Dr. Surinder Kapur, Chairman and Managing Director of Sona Koyo Steeling Systems Ltd. and founder of Prof. Shiba's Learning Community in India

UCAL has been associated with Prof. Shoji Shiba in learning the process of breakthrough management. It has been an exhilarating experience for the young managers of our company. Our organization has greatly benefited by practicing various concepts and methods taught by Prof. Shiba, and we have become more able to face the uncertainties of the future.

- Mr. K. Jaykar, Managing Director, UCAL Fuel Systems Ltd.

Prof. Shiba's breakthrough management principles, concepts and tools are deceptively simple and yet incredibly powerful. His workshops have transformed TechNova's culture beyond recognition. Personally, I have gained incredible insights into my own leadership style. With his extraordinary teaching style, we rapidly learned to unlearn, look through the invisible and the unknown, and sense the future.

- Mr. Pranav Parikh

Also by Prof. Shoji Shiba and available from CII



ISBN 978-81-903364-1-0

4.1 Our first publisher and printer

Our intention had been to publish the book ourselves in the United States (before publishing it anywhere else in the world) using print-on-demand (e.g., Lightning Source). To this end, I had always planned to prepare a ready-to-print PDF file of the manuscript.

However, suddenly the opportunity arose to first publish the book in India using an industry group my co-author worked with there as the nominal publisher and distributor, and this industry group had a recommended printer in New Delhi (I had previously been unable to find the equivalent of Lightning Source for print-on-demand in India). I immediately began a major email correspondence with the publisher and printer to nail down a variety of mundane but important issues:

- the format of the publisher’s name and logo on the title page and cover
- the publisher’s information on the back of the title page including an ISBN number
- what promotional text and images the publisher wanted on the back cover of the book
- whether the printer could print from a PDF file of the whole manuscript doing the page imposition itself, or would I have to learn algorithms and find software tools to create big PDF pages with 4 or 8 book size pages on each big page that would later be folded and cut into book pages in the correct order: the printer would do the imposition but wanted to work from a PostScript file which I planned to obtain by saving my PDF file as a PostScript from Adobe Acrobat
- would I have to use a different page block size for India (where I thought perhaps they used different size book pages than the frequently used U.S. standard page size of 9×6 inches), or could I find a compromise text block size that worked in both the United States and India (William Adams gave me guidance on a compromise text block size): it turned out the the problem was non-existent as the printer in India is happy to print on 9×6 inch pages, and I used the `\url{geometry}` package to place text blocks of the correct size (within 7.75×5 inches) on the correct size pages.
- could I send my cover design to the printer as an Illustrator file (and for what version of Illustrator) that the printer could use to make final adjustments
- did I need to convert some of my photographic images which were in color to gray scale (there were to be no color plates in the book), or would that conversion automatically happen as part of the B&W printing process: they asked me to do the conversion

- when I sent the printer my ready-to-print PostScript file and many Illustrator files, did the printer want those on multiple CDs or one DVD: we agreed the printer would try to download the PostScript file of the book and the Illustrator file of cover art from my website, and that I would send a couple of CDs containing a gigabyte of backup files (mostly images) by air courier

In parallel with sorting out the issues with the Indian publisher and printer, my co-author and I worked on the design of the outside cover. After several iterations with Adobe Illustrator, I produced a design that satisfied us both.

4.2 Refining the line and page layout

I have always disliked how \LaTeX , left to its own devices, feels free to stretch the inter-paragraph spacing. Steve Peter told me that the first step should be to give the command

```
\parskip=0pt
```

to disallow insertion of inter-paragraph vertical space. Thus, vertical space got automatically added only around floats, titles, thought breaks (www.tug.org/pracjourn/2005-4/walden/), and so forth. (At PracTeXo6, TPJ editorial board member Kaveh Bazargan gave a wonderful paper on a set of macros his company has developed for controlling vertical stretch, thus facilitating lining up the lines of text in parallel columns or on facing pages.)

I next read section 4.5 of the second edition of *The \LaTeX Companion* on visual layout to avoid pages with too much vertical space, facing pages with obviously different numbers of lines, and the like. Based on the ideas presented there, I added the following commands to my class file

```
\newcommand{\Lnewpage}{\newpage}
\newcommand{\Lpushlines}[1]{\enlargethispage{-#1\baselineskip}}
\newcommand{\Lpulllines}[1]{\enlargethispage{#1\baselineskip}}
\newcommand{\Lhyphen}{\ -}
\newcommand{\Lhspace}[1]{\hspace{#1}}
\newcommand{\Lvspace}[1]{\vspace{#1}}
```

The $\backslash L$ at the beginning of each new command (L standing for Layout) is to let me easily find all uses of these commands to remove them for a later different page layout of pages of the book.

Probably one page in six needed manual adjustment using these new commands. Viewing facing PDF pages side by side in my Acrobat Reader was key to seeing where such

adjustments were needed and trying out various possibilities for fixing things (plus lots of recompiles of each chapter). Other techniques that proved useful were (a) slight adjustments of figure and table sizes (an invisible tiny fraction of an inch change sometimes made things lots better), and (b) encouraging floats of figures and tables in some cases to be placed at the bottoms of facing pages to avoid having to balance full pages of text on facing pages.

In a dozen or so places in my 280 page book, I had to rewrite a part of a sentence to avoid problem lines. I also had to put in a few specific hyphens that T_EX's automatic hyphenator couldn't find, particularly in proper names. I also spotted a few more instances where I thought kerning was required, I added explicit kerning in these instances, and (to mark the spots for later removal of the explicit kerning) I used my \L convention:

```
\newcommand{\LVught}{V\kern-1pt{}ught}  
\newcommand{\LVerschuren}{V\kern-1pt{}erschuren}
```

I reproofread each chapter before I made the changes described above and checked the changes after I finished each chapter.

A book with all text might need fewer adjustments, but my book had perhaps 100 figures in 280 pages, lots of subtitles, many block quotes, and many lists that all had to be well placed on a page, and it seems to me that this made the layout harder most of the time. I did all the manual adjustments by hit and miss using what looked good to me as my only standard. Next time I first will read some theory about proper page layout.

5 Printing

The printer in India wanted a PostScript file of the manuscript of the book from which they could do imposition. I obtained this by opening the PDF of the manuscript in Adobe Acrobat and saving it to .ps format which I posted to my website for the printer to download. I also saved this on a CD and all of the source (text and image) files on a DVD and mailed them to the printer via international air courier for backup.

Sending the cover art to the printer was more difficult, however. They could not access the .ai file I provided out of Illustrator even though they purportedly were using the same version I was (CS2). They decided it would be best for me to send them a PostScript file. Unfortunately, as far as I know, Illustrator outputs .eps files and not .ps files. (At PracTeX06 Jonathan Kew and Peter Flynn told me I can convert an .eps file without previews to a .ps file by removing the e from the file name extension.) I tried converting a PDF

file out of Illustrator to PostScript via Acrobat, but I couldn't make this work for some reason (all the art content was lost somewhere along the conversion path, or the sizing information was confused).

Eventually, I converted the Illustrator EPS file to a PDF using `epstopdf` (as I did for all the rest of my illustrations), wrote a nine line \LaTeX program that set up the geometry of the page size I needed for the cover (front, spine, and back as one image), and did an `\includegraphics` of the .pdf image. I then compiled with `pdf \LaTeX` , opened the PDF output in Acrobat, and saved it in .ps format; this worked and indicated to me that the problem was in using the Illustrator produced PDF. Therefore, I ran a test of converting the EPS output of Illustrator to PDF via `epstopdf`, then opening the PDF in Acrobat, and then saving to .ps. This apparently worked.

In the end, I provided .eps, .ps, and .pdf files of the cover to the printer; I'm not sure which one the printer used.

6 North American and European distribution

With the book published in India, my next problem will be printing and distribution in North American and Europe. Various issues come to mind:

- Who should the publisher-in-name be (I will be the actual publisher)? Should I ask the Indian publisher to keep its name on the book for this hemisphere, should I seek another industry consortium, or should I make up my own publisher name? There are also related ISBN number issues.
- Will I have to resize the text block (and reformat the pages), or will I want to improve the overall typesetting in light of what I see in the Indian printing? In particular, will the Minion font I chose for my main text reproduce well using the printing technologies of print-on-demand printers?
- I believe that by printing via a print-on-demand company (e.g., Lightning source), I will have distribution (via Ingram) to U.S. on-line book stores (such as Amazon.com and bn.com). Will connecting to Lightning Source's UK group also provide distribution via on-line book stores in Europe?
- I have the option of selling books direct from my website (www.walden-family.com/breakthrough) in the United States. What parallel arrangement might I make for Europe?

(Elizabeth Dearborn's presentation at PracTeXo6 on self-publishing a book provided inspiration that I can manage the logistics of publishing my own book.)

While these last book-publishing steps are mostly not T_EX issues, I will give a brief final report on what transpired in my next column.

Acknowledgements

My copy editor, Jay Howland, helped me settle on a number of design points for my book. Steve Peter helped me extensively as I prepared the book to go to the printer. William Adams answered more than one question and suggested a number of changes to a draft version of the manuscript. Karl Berry gave me comments on another draft. Peter Flom, Hans Hagen, Steve Peter, Will Robertson, and particularly Barbara Beeton gave me many useful comments on the nearly final draft.

Biographical note

David Walden is retired after a career as an engineer, engineering manager, and general manager involved with research and development of computer and other high tech systems. More history is at www.walden-family.com/dave.

[Journal
home page](#)
[General
information](#)
[Submit an
item](#)
[Download
style files](#)
[Copyright](#)
[Contact us](#)

THE PracTeX Journal



Book Review: *Formatting Information -- A beginner's introduction to typesetting with LaTeX* by Peter Flynn

D.V.L.K.D.P. Venugopal

Abstract

The reviewer sees Peter Flynn's *Formatting Information* as a more practical introduction to LaTeX than many other popular introductions.

Mr. Venugopal is a stenographer by profession. He has been using computers since the two floppy PC stage. A former lover of WordPerfect, his first encounter with TeX/LaTeX was in the year 2000. At present exploring various packages, experimenting with Omega, MetaPost and MetaFont and teaching and/or explaining advantages of TeX/LaTeX over wordprocessors are his hobbies. You can contact him at `venugopal_duvvuri@rediffmail.com`

- [PDF version of paper](#)
- [Comment on this paper](#)
- [Send submission idea to editor](#)

Page generated August 19, 2006 ; [TUG home page](#); [search](#); [contact webmaster](#).

Book Review

Formatting Information: A beginner's introduction to typesetting with L^AT_EX by Peter Flynn

D.V.L.K.D.P. Venugopal

Many excellent books are available introducing L^AT_EX to beginners. Some are freely available on the net like *The (Not So) Short Introduction to L^AT_EX2_ε: L^AT_EX2_ε in 131 Minutes* by Oetiker et al.¹ whereas some are nicely published like *A Guide to L^AT_EX* by Kopka and Daly.² My favorite introduction is Peter Flynn's *Formatting Information*³ which has more emphasis on the beginning user's overall set of problems (particularly in chapters 1, 2, and 10) than some books do.

In particular, many introductory books focus on L^AT_EX and more or less ignore the rest of the beginner's development environment. This is understandable: Which of the many editors and other aspects of the user's development environment should a general L^AT_EX introduction select?

While learning L^AT_EX I had to struggle a lot, and I am sure many other beginners have similar problems. I was unaware of the various editors like emacs or vi with which one can easily handle typing, compiling and viewing the document without leaving the editor. Generic instructions along the lines of

Type your text in plain vanilla editor. Save it with .tex extension.
Then run L^AT_EX on it with `latex somefile` and you will get the output `somefile.dvi`. View it with a DVI viewer.

¹<http://www.ctan.org/tex-archive/info/lshort>

²Helmut Kopka and Patrick W. Daly, *A Guide to L^AT_EX*, fourth edition, Addison-Wesley, 2004.

³<http://www.tug.org/tex-archive/info/beginlatex/beginlatex-3.6.pdf>

were not sufficient for me. Half the struggle in the initial stages of learning L^AT_EX doesn't have much to do with L^AT_EX itself. If information on the user's overall development environment is not provided, it may leave the user confused and inclined to drop the idea of learning T_EX or L^AT_EX.

It was through the postings to the Indian TUG that I learned about emacs and its AucTeX mode. I happened to see Peter Flynn's book, *Formatting Information: A beginner's introduction to typesetting with L^AT_EX*, in the year 2003 (version 3.2) and was attracted to its lucid style of presenting L^AT_EX to the beginners. It not only introduced the concepts of integrated development environments available for various platforms but also gave important information about installing new packages and fonts and modifying macros. Later *TUGBoat* dedicated its November 2003 issue for this book (version 3.4).⁴

The earlier name of Flynn's book was *Beginner's L^AT_EX*, and it started its journey as lecture notes for a 10 session L^AT_EX introductory course. The earliest edition I could obtain was version 2.2 (December, 2001) with 75 pages. As the years passed and at the request of readers, Peter Flynn added more content to the book. Recently I stumbled upon version 3.6 of this book which has grown to an awesome 275 pages.⁵ The rest of this note applies to version 3.6.

The book is divided into 10 sessions or chapters. By reading and practicing the exercises given in it one can gain enough confidence to tackle L^AT_EX.

The first chapter contains discussion about various T_EX distributions available, obtaining them and installation instructions for various platforms. (In earlier editions this chapter contained details about the various editors available, but in the present edition this information was moved to second chapter.)

The second chapter deals with creating the T_EX documents, the concept of markup and the various popular editors that are available including LyX, T_EXshell, Winshell, T_EXnicCenter, WinEdt and emacs along with screen shots. The concepts of typing the document and the various special characters are also discussed.

Document structure along with the concepts of documentclass and its options, various environments, commands for obtaining the Table of Contents, List of Tables, etc. are handled in Chapter 3. This chapter also contains information on methods of changing the abstract name, adding entries to table of contents, changing the spacing between paragraphs, and paragraph indentation.

⁴<http://www.tug.org/TUGboat/Contents/contents23-2.html>

⁵<http://www.tug.org/tex-archive/info/beginlatex/beginlatex-3.6.pdf>

Chapter 4 deals with typesetting, viewing and printing of the documents. Various error messages, their meaning and how to handle them are explained in this chapter.

The concept of adding on features using *packages*, downloading *packages* from CTAN and installing them are discussed in Chapter 5. The concepts of the T_EX Directory System (TDS), various types of files used by T_EX like .sty, .cls, .fd, .mf and where to place all these files are well illustrated. This chapter also lists various places where a user in need of help can find desired information, like T_EX Users Group (TUG) and various mailing lists.

Chapter 6 discusses including graphics (figures and images) in the document, bulleted and numbered lists, tables, verbatim text, boxed matters, panels and side-bars.

Chapter 7 introduces the reader to typesetting quotations, footnotes, marginal notes, cross references, bibliographic references and the BibT_EX. Creating bibliographic data bases for BibT_EX and the freely available graphical interfaces to the BibT_EX databases were also discussed. (Please note that there is a BibDB programme that is freely available for Windows and DOS platforms.) Creating multi-column documents, index and glossary were also discussed in this chapter.

Changing the layout of the document and using fonts in L^AT_EX documents is nicely handled in Chapter 8. The various fonts available in METAFONT and the freely available Adobe and X Consortium fonts along with the name of the family and the necessary packages, the various size changing commands, and changing the colour of the text were also illustrated. Installing METAFONT fonts and postscript fonts, creating font definition files are also discussed in this chapter.

One of the most powerful feature of L^AT_EX, programmability, is discussed in Chapter 9. Creating new commands, changing previously defined macros, creating macros with arguments are nicely introduced to the readers with examples.

The last chapter discusses the compatibility with other systems of typesetting like Quark XPress, Pagemaker, FrameMaker etc. The various commercial and shareware software available to convert L^AT_EX to other formats (MS Word, WordPerfect) and vice versa are also discussed. Because typically it is not possible to convert from one format to another format completely cleanly, the book only deals with how the pain can be decreased and not eliminated.

In addition to these ten chapters, the book has three Appendices dealing with configuring T_EX search paths, how to become a member of the T_EX Users Group

and its benefits, and the ASCII character set. A fourth appendix contains the GNU Free Documentation License under which author Flynn has generously made this book available.

To help the reader use the book, it has a detailed table of contents (down to the subsubsection level), another table listing the page numbers of exercises that show how to undertake 20 common tasks, another table listing the page numbers of several useful charts, and a fairly thorough index.

Since Flynn's book lacks deep discussion about typesetting mathematics and other matters of document formatting, I recommend its use in conjunction with the *L^AT_EX Tutorials*⁶ and Herbert Voß's *Math mode*.⁷ I greatly appreciate having these three books available in PDF format. I keep them on my computer's desktop and, whenever I am in doubt about something, I just open one of these books, read the appropriate discussion and use the code. Another advantage of these PDF books over the printed books is that the authors regularly modify and improve the content. (Another book which is undergoing regular update is Oetiker's *The (Not So) Short Introduction to L^AT_EX2_ε*.)

For my L^AT_EX courses I distributed the earlier version (version 3.4) of Flynn's book along with the *L^AT_EX Tutorials* from the Indian TUG Group. The books complement each other and the students easily begin to grasp L^AT_EX.

Overall I believe Peter Flynn's book is a must read for every L^AT_EX beginner, and we should thank Flynn for giving us such a fine introduction to the practical use of L^AT_EX.

⁶<http://www.tug.org/tutorials/tugindia/>

⁷<http://www.tug.org/tex-archive/info/math/voss/mathmode/Mathmode.pdf>

[Journal
home page](#)
[General
information](#)
[Submit an
item](#)
[Download
style files](#)
[Copyright](#)
[Contact us](#)

THE PracTeX Journal



Ask Nelly: What are the differences among MiKTeX, ProTeXt, and LaTeX? How do I add change bars?

The Editors

Abstract

Ask Nelly is a question and answer column. Nelly is the quiet person who sits at the back corner desk, who knows a lot, and when asked any question is always ready with a patient answer. If Nelly doesn't know the answer, Nelly will know an expert who has the answer. Feel free to [Ask Nelly](#) about any aspect of LaTeX, TeX, Context, etc.

- [Comment on this paper](#)
- [Send submission idea to editor](#)

TpJ

Q: Dear Nelly: This is my first contact to the TeX World and I cannot understand what is the difference among MiKTeX, ProTeXt, LaTeX. Can you help me?

A: MiKTeX is a distribution of TeX including the basic TeX typesetting engine plus many other things. There are other commercial and free distributions of TeX, each containing many of the same things (including LaTeX) and a some distinguishing things.

ProTeXt is the MiKTeX distribution plus a nice PDF guide of the installation procedure plus a couple of editor options that are not part of the MiKTeX distribution.

LaTeX is a set of TeX macros (commands) (such a set is called a "format" in the TeX world) that makes a particular set of assumptions about what a book, report, letter, etc., look like;

many people find LaTeX more appropriate to use than the "plain TeX" format of the basic underlying TeX typesetting engine; LaTeX also has provisions for modifying many of these LaTeX assumptions.

The MiKTeX distribution also comes with other alternatives to the LaTeX format (modifications to LaTeX or built directly on basic TeX) that offer different starting assumptions about what various documents look like and the level of detail the user wants to give commands to TeX.

The above question was answered by **Dave Walden**, a member of the editorial board of this journal.

TPJ

Q: Dear Nelly: Have you covered how to add vertical lines to the margins to indicate areas of change, editing, revision, etc. In resubmitting grants I need to tell the reviewers what areas have changes. I don't want to italicize, bold, or underline the text.

A: The changebar package described on pages 189-191 of the second edition of *The LaTeX Companion* may help you (I always go to the index of this book first when looking for a new capability). Having found the package in the book, I also found (by putting `\RequirePackage{changebar}` in the preamble of a document and trying to compile the document) that the changebar package is already included in the ProTeXt distribution of TeX that I use; thus I assume many or most other distributions also include it. I then also found a document on the changebar package already in my computer by doing a search of my `\texmf` file hierarchy for "changebar"; it found `changebar.dvi` among several other files, I clicked on `changebar.dvi`, and the documentation opened in my previewer window.

You might also look in the FAQ, i.e., at <http://www.tex.ac.uk/cgi-bin/texfaq2html?label=changebars>

The above question also was answered by **Dave Walden**.

TPJ

[Journal](#)
[home page](#)
[General](#)
[information](#)
[Submit an](#)
[item](#)
[Download](#)
[style files](#)
[Copyright](#)
[Contact us](#)

THE PracTeX Journal



Distractions: LaTeX Wordplay — 3 crosswords; math font quiz answers

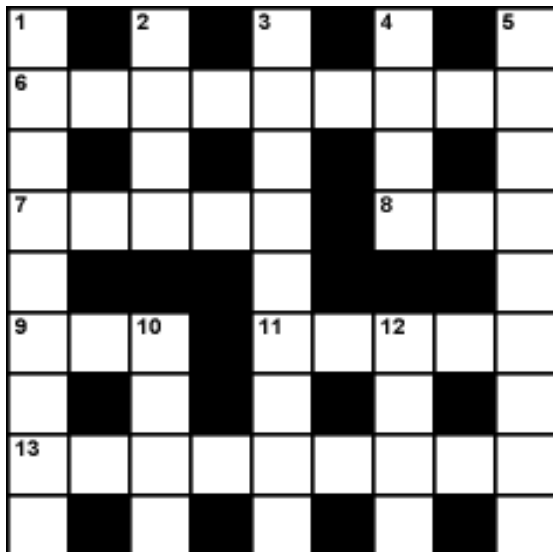
The Editors

- [Comment on this paper](#)
- [Send submission idea to editor](#)

[Crossword puzzles](#)

[Math font quiz answers](#)

LaTeX Wordplay — Three LaTeX theme crosswords



In order of difficulty, easiest first:

[Joy & Dave Morris's puzzle](#)

[Arthur O'Dwyer's puzzle, with medium-difficulty clues](#)

... [the same puzzle with hard clues](#)

[Bob Newell's Cryptic crossword](#)

(Solutions below)

A few months ago the movie [Wordplay](#), about crossword puzzles and the people who solve them, was released. It's a great little film, even if you're not a puzzle fan. In the movie one of the great puzzle makers, [Merl Reagle](#), showed how he compiles a puzzle with pencil and paper. (Crossword makers are called "compilers", "setters", or even "cruciverbalists".)

It would have been nice to have Merl Reagle compile a LaTeX theme puzzle for the PracTeX Journal, but since he probably thinks of LaTeX as a pliable plastic substance the result might have been different than expected ;-). Fortunately, several LaTeX users compiled some excellent puzzles for your enjoyment.

Joy and Dave Morris, both math professors at the [University of Lethbridge, Canada](#), created a large puzzle full of LaTeX and math-related answers and clues. This puzzle is their first try at compiling a crossword, and they used the LaTeX `crosswr` package to typeset it.

Arthur J. O'Dwyer of Carnegie-Mellon, who compiles puzzles for the student newspaper and has written some [crossword-making software](#), used his software to make a challenging LaTeX-themed puzzle. His C program, `xword-typeset`, generates a crossword in the form of a LaTeX file which uses the `cwpuzzle` package. The puzzle Arthur originally sent had really tough clues, and he allowed them to be edited down provided he could make a second set of *really* tough clues. These two versions of his puzzle are available at the top of this page.

Bob Newell, a [games person](#) who has also written some crossword software and who once participated in the American Crossword Puzzle Tournament (featured in the movie *Wordplay*), made a cryptic or British-style crossword. (If you're not familiar with this type of puzzle, see this [cryptic crossword](#) explanation.) Bob is an avid cryptic solver, but since this was his first try at compiling a cryptic crossword he asked for suggestions to improve the clues. Several of us helped with this and it turned out to be a lot of fun. It seemed that each time someone came up with a pretty good clue, someone else would then improve it, then someone else would improve it a little more, ...

Peter Flom, a Scrabble® expert, helped with clues. **Gerd Neugebauer**, the author of the `cwpuzzle` package, made a heavily TeX and LaTeX-themed puzzle which will appear in a future Distractions column.

Hope you enjoy solving the puzzles.

[Solution to Joy & Dave Morris's puzzle](#)

[Solution to Arthur O'Dwyer's puzzle, both versions](#)

[Solution to Bob Newell's Cryptic crossword](#)

Math font quiz answers

In the 2006-1 "fonts" issue Michael Spivak supplied a challenging [math font quiz](#). No one submitted an answer by the deadline so we extended the contest and offered a prize to the person who answered three or more questions correctly. **Harri Haanpaa** of the Helsinki University of Technology answered several of the questions correctly, and we will be sending him a copy of the MathTime Professional II Lite fonts.

[Answers](#) to the math font quiz from [Issue 2006-1](#)

Page generated August 19, 2006 ; [TUG home page](#); [search](#); [contact webmaster](#).